# A High Order Overset Flux Reconstruction Method for Dynamic Moving Grids



Zhaowen Duan

Submitted to the graduate degree program in Department of Aerospace Engineering, University of Kansas and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Committee members

ZJ Wang, Chairperson

Saeed Farokhi

Ray Taghavi

Huixuan Wu

Xuemin Tu

Date defended: December 13, 2019

The Dissertation Committee for Zhaowen Duan certifies
that this is the approved version of the following dissertation :

A High Order Overset Flux Reconstruction Method for Dynamic Moving Grids

ZJ Wang, Chairperson

Date approved: _____

# Abstract

Overset meshes have a unique advantage in handling moving boundary problems as remeshing is often unnecessary. Recently, overset Cartesian and strand meshes were used successfully to compute complex flow over rotorcraft. Although it is quite straightforward to deploy a high-order finite difference method on the Cartesian mesh, the near-body solver for the strand mesh is often limited to second order accuracy. In the present work of this dissertation, we develop a high-order FR/CPR solver, hpMusic, on both the near-body and background grids, and extend it to handle moving boundary problems. The solver is also extended to sliding meshes, which can be considered a special case of overset meshes. The use of sliding meshes can often simplify the treatment of moving boundary problems with simple translational and rotational motions. Two different approaches to handle the overset interfaces are evaluated for accuracy, efficiency and robustness. Accuracy studies are carried out and the designed order of accuracy is obtained for both inviscid and viscous flows. Steady and unsteady flow problems are solved on stationary overset meshes. The results agree well with those in the literature and from experiments. A turbine blade under the wake of moving cylinders is simulated using sliding meshes. The flow structures are compared with those without moving cylinders. The solver is then tested for moving overset meshes with a benchmark dynamic airfoil problem from the 4th International Workshop on High-Order CFD Methods. Hp-convergent results are obtained and compared with those from other groups. Finally flow over a hovering rotor is simulated to compare with experimental data. In this case, the present high-order solver is capable of generating and propagating tip vortices with high resolution. Good agreement is achieved with experimental data in tip vortex core size, location, and the swirl velocity at 3rd order accuracy.

# Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor Dr. Z.J. Wang for his support and encouragement. He is always willing to answer my questions and point out the directions for my research work. I have also improved a lot in coding skills from his expertise so that I can efficiently convert my ideas into CFD codes. I would like to give thanks to my comittee members Dr. Saeed Farokhi, Dr. Ray Taghavi, Dr. Xuemin Tu, and Dr. Huixuan Wu for their guidance and support in various roles. Special thanks goes to Dr. Xuemin Tu for her teaching of numerical methods for solving linear systems and partial differential equations in various courses. I would like also thank Dr. Zhongquan Zheng for helping me in fluid dynamics courses. These courses give me a strong background in fluid dynamics and advanced numerical methods.

Finally I want to thank my family and friends who have been unbelievably supportive throughout my academic career, and specifically throughout my Ph.D. To my wife Min and three lovely kids An, Heyi and Hemei: I could not have completed this journey without your love and understanding; you have sacrificed so much during the process that has led to this dissertation. Thank you!

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Motivation

Computational fluid dynamics (CFD) simulations of flow over rotorcraft are challenging because of the complex physics associated with vortex dominated turbulent flow and rotating rotors. Vortices are generated at the tip of a rotor blade and interact with other blades and the fuselage resulting in complex blade-vortex interactions (Rockwell, 1998) and vortex wake interactions (Strawn et al., 1999). Besides the fluid dynamics, rotorcraft flow is multidisciplinary, requiring the solution of moving-body aerodynamics coupled with structural dynamics for rotor blade deformations, vehicle flight dynamics and controls (Strawn et al., 2006; Lim & Strawn, 2008; Holst & Pulliam, 2010). Most CFD simulations use meshes which discretize the entire flow domain with a single non-overlapping mesh. However, moving boundary problems are difficult to simulate with non-overlapping meshes because re-meshing is often needed whenever the mesh quality becomes low. The relative motion between rotor blades and the fuselage of rotorcraft naturally leads to the use of overset mesh methods. Overset meshes, or the so-called Chimera meshes, discretize a flow domain with multiple overlapping grids. The grids are allowed to move independently while preserving mesh quality without re-meshing. When numerical methods are implemented on overset meshes, flow solution data is communicated between meshes at their overlapping region. In the simulations of moving rotors or wind turbines, the overset framework often includes near body meshes with motions imposed or computed based on the forces, and a (Cartesian) background mesh which is stationary. One of the state-of-the-art computational tools for solving flow over rotorcraft is the Helios computational platform (Sankaran et al., 2011). In the Helios platform, overset block-

adaptive Cartesian and near body unstructured meshes including a strand mesh (Katz et al., 2010; Sitaraman et al., 2017) were employed to handle moving bodies. Generally a near body 2nd-order unstructured grid solver is coupled with a high-order Cartesian grid solver to compute the unsteady aerodynamic flow field (Lakshminarayan et al., 2017) . Due to the complexity of such flow problems the accuracy of the computational solution is important. The primary objective of the present study is to develop and demonstrate a high-order solver for both the near body and background meshes to enhance the accuracy and efficiency of rotorcraft flow computations. This high-order solver should also be able to handle relative motions of overset meshes.

## 1.2 High-Order Methods

High order methods capable of solving the Navier-Stokes equations on unstructured grids are a major development in computational fluid dynamics in the last two decades. The order of accuracy of a method refers to the convergence rate of the solution with respect to the mesh size when the grid spacing $h \to 0$. For most commercial or production codes, the order of accuracy is 2nd order, i.e. $O(h^2)$. A method (or a flow solver) is considered high-order when the convergence rate is 3rd-order or higher, i.e. $O(h^p)$ with $p \geq 3$. Comparing with 1st or 2nd order methods, high-order methods can deliver higher accuracy with less CPU time for scale-resolving simulations, as demonstrated in the multiple International Workshops on High-Order CFD Methods (Wang et al., 2013). The following sub-sections review some common high-order methods for CFD simulations.

### 1.2.1 Finite Difference Methods

The finite difference methods approximate the differential operators in a PDE system through a weighted summation of solutions at discrete points in a domain(Thomas, 2013), resulting in a system of algebraic equations with the same number of unknowns as the the number of discrete points. This method is derived by performing a Taylor series expansion. The order of accuracy depends on the choice of the neighboring points, which is known as the stencil. The 1st-order finite difference

method for CFD simulations can be traced back to the late 50s, developed by Godunov (Godunov, 1959). The extension of this method is known as Monotonic Upstream-centered Scheme for Conservation Laws (MUSCL) (Van Leer, 1979). The MUSCL scheme has become the industry standard for many 2nd- and 3rd-order flow solvers. The stability of the MUSCL scheme relies on non-linear limiters to eliminate the oscillation of numerical solutions, which is known as the Gibbs phenomenon (Arfken, 1985). A higher-order finite difference scheme, the Essentially Non-oscillatory (ENO) scheme, was introduced later by Harten et al (Harten et al., 1987). The ENO scheme uses the reconstruction of high-order non-oscillatory polynomials along with limiters and averaged values of neighboring nodes to achieve high-order accuracy. The order of accuracy of the ENO scheme was then improved by the Weighted Essential Non-oscillatory (WENO) scheme (Liu et al., 1994). In finite difference schemes, the number of nodes in a stencil is usually one greater than the order of the scheme, e.g. a 4th-order scheme requires five nodes inside the stencil. For high-order schemes, the large size of the stencils impose restrictions on grid quality for stability reasons. In addition, they are complicated to implementation in parallel computing. An alternative approach to reduce the stencil size is the compact difference scheme (Lele, 1992). In this scheme the size of the stencil is reduced by implicitly computing the numerical derivatives. Due to the implicit formulation, the scheme is restricted to structured meshes. In addition, the cost of reducing the stencil size is the sensitivity of the scheme to grid irregularities (Visbal & Gaitonde, 2002). In practice, the grid is required to be nearly $C_1$ continuous along grid lines and the cell stretching is limited to 10% for stability reasons.

### 1.2.2 Finite Volume Methods

The finite volume methods start with the integral form of the governing conservation laws. Then the volume integral is transformed into surface integrals using Gauss's rule. The domain is decomposed into a set of control volumes. The numerical solution at each control volume is taken to be the averaged value over that volume. The solution distributions are reconstructed from the averaged values. Then the common flux at the interface between two control volumes is approxi-

mated by an upwind flux (Van Leer, 1985) or computed with a Riemann solvers (Toro, 2013). The surface integrals are approximated by quadrature formulas.

Since there is little restriction on the shape of control volumes, finite volume methods are suitable for any mesh types. Flow solutions with complex geometries can be computed on unstructured meshes. Due to these properties, 2nd-order finite volume methods are perhaps the most widely used methods for flow solvers in industry. Similar to high-order finite difference schemes, it is possible to reconstruct high-order polynomials based on the solutions of neighboring cells. Therefore, high-order accuracy can be achieved. However, larger stencils are needed for high-order finite volume schemes, especially in three spatial dimensions (Delanaye & Liu, 1999). This leads to an increase in complexity and in the amount of data to be communicated in a parallel implementation. It also increases the difficulty in implementing implicit iterative algorithms. (Mavriplis, 2002).

### 1.2.3 Finite Element Methods

Finite element methods were originally developed for structural mechanics problems (Turner, 1956; Argyris & Kelsey, 1960; Clough, 1960), and then extended to solve systems of PDEs(Zienkiewicz et al., 1977; Hughes, 2012). In a finite element method, basis functions (also called shape functions) are defined locally on each element of the domain. The field solutions on the element are approximated by a linear combination of these shape functions. Polynomials are the most commonly used shape functions. The order of accuracy increases with the degree of the polynomials. Similar to the finite volume method, the governing equations are solved in integral form, which is obtained by multiplying the governing equations by a set of test functions and integrating over the element. The integrals are then converted to their weak form through integration by parts. The weak form consists integrals over both the volume and the surfaces of the element. The volume integral can be directly evaluated if the test functions are given. For instance, the shape functions can be used as the test functions, resulting in a Galerkin method. The linear combinations of these integrals can therefore be written as a form of matrix-vector multiplication. If the numerical solution is continuous across element interfaces, this method is known as the continuous Galerkin

(CG) method. Otherwise, the methods is called the discontinuous Galerkin (DG) method (Arnold et al., 2002). In a DG method for CFD simulations (Klaij et al., 2006), similar to the finite volume method, the flux at the interfaces is replaced by a common flux, which is computed by solving a Riemann problem (Toro, 2013; Van Leer, 1985).

Since Galerkin methods use the integral form, it is suitable for both structured and unstructured meshes. Moreover, the locally defined piece-wise high-order polynomials naturally lead to high-order accuracy without increasing the stencil size. In fact, the DG stencil only relies on the immediate neighboring elements. This make the DG method a compact method and efficient for a parallel implementation. In a CG finite element approach, an element is globally coupled to all other elements. However, in a DG method, an element is only explicitly coupled to its immediate face neighbors. This compactness allows the DG solution to be marched in time with explicit Runge-Kutta schemes. In an implicit scheme, the implicit matrices are sparse for a DG method. These properties lead the DG method to be the most widely used high-order method on unstructured meshes. (Bassi & Rebay, 1997; Cockburn et al., 2000; Shu, 2014).

Another popular finite element based high-order method is the flux reconstruction (FR) or the correction procedure via reconstruction (CPR) method (Huynh, 2007). It shares the same solution space with the DG method, and the derivation of this method is very similar to the DG method. However, the common fluxes at the element interfaces are used to generate a correction field to the numerical solutions inside the element. The integral form is then converted to a equivalent differential form. The derivatives are computed by the linear combination of the differentials of the shape functions, which are defined on a set of solution points located inside the element. The correction field increases the degree of approximate polynomials by one, which cancels the one degree lost due to operating differentials. The FR/CPR method was originally developed by Huynh (Huynh, 2007), and later extended to simplex and mixed unstructured meshes (Wang & Gao, 2009; Haga et al., 2011). Since the method shares most advantages of the DG method and is more straightforward to implement, it has attracted considerable attention recently (Vincent et al., 2011). We will introduce the FR/CPR method in more details in the following chapter.

The well known high-order methods other than the above mentioned discontinuous finite element are the spectral volume (Wang & Liu, 2002) and spectral different (Liu et al., 2006) methods. In the spectral volume method, each cell of the mesh is subdivided into multiple control volumes which are called sub-cells. Similar to the finite volume method, the approximate solution is stored as a mean value for each sub-cell. The solution inside the cell is represented as a polynomial that is reconstructed from these mean values of the sub-cells. The mean values of the sub-cells are updated using the same principals as a traditional finite volume method. The fluxes on the interfaces between sub-cells within the same cell are directly evaluated from the reconstructed polynomial. However, the fluxes on the interface between two cells are computed by Riemann solvers. Appropriate quadrature rules are required to compute the integrations of the fluxes on the interfaces of the sub-cells. In higher dimensions, using the quadrature increases the computational cost dramatically. The spectral volume method shares many features with the DG method. The solutions are approximated as piecewise polynomials in a cell for both methods. They are also conservative, compact, suitable for parallelization, and can handle complex geometries. Numerical experiments have shown that the DG method generally achieves lower magnitude errors, while the spectral volume method can be updated with larger time steps (Wang & Liu, 2002; Zhang & Shu, 2005).

The spectral difference method was originally proposed by Liu et al. (Liu et al., 2006) for solving wave equations on triangular grids. It was then extended to 2D Euler equations (Wang et al., 2007) and further to three-dimensional Navier–Stokes equations on hexahedral unstructured meshes (Sun et al., 2007). In the spectral difference method, the solution is represented by polynomials reconstructed from the values at a set of points defined inside a cell. The points are normally chosen to be at quadrature points that will approximate the volume integral over the cell to the desired order of accuracy. The solutions are updated using the differential form of the governing equations by approximating the flux derivatives at those solution points. The derivatives are computed from the polynomial represented fluxes. These polynomials are reconstructed from the fluxes at certain points, which are also known as flux points. The flux points are often different from the solution points, and a big portion of them are located at the surfaces of a cell. Similar to

the DG method and the spectral volume method, the discontinuous fluxes at the interfaces of cells are replaced by the common fluxes computed by Riemann solvers. Since no integral is explicitly involved, the spectral difference method is more straightforward than DG and spectral volume. Due to the piecewise polynomial representing of the solutions, the spectral difference method is compact and works well with unstructured meshes. However, comparing to the FR/CPR method, the little coincidence between the flux points and the solution points at the cell interior of this method introduces complexity. Furthermore, it requires more flux points reconstructing one-degree higher polynomials than solution polynomials to compensate the degree lost due to the derivative. This leads to more computational cost than the FR/CPR method. It is also reported that weak instability was found when implementing high-order spectral difference method on triangular grids (Van den Abeele et al., 2008).

## 1.3 Overset Mesh Approaches

The first overset method for solving CFD problems was introduced in 1980's (Steger et al., 1983). It used a finite difference method to solve the Euler equations on overset structured grids. Various overset methods were then developed for different flow regimes and applications. In 1990's overset finite volume methods were also developed for both the Euler equations and Navier-Stokes (NS) equations (Benek et al., 1985; Pärt-Enander & Sjögreen, 1994; Henshaw, 1994; Fujii, 1995; Wang, 1998; Brown et al., 1999). Most of these methods are for structured meshes. Then, in the early 2000's, overset methods were developed for unstructured meshes (Nakahashi et al., 2000; Loehner et al., 2001; Rogers et al., 2003). Tools like Structured Unstructured and Generalized overset Grid Assembler (SUGGAR) allow different solvers and mesh types (Noack, 2005). Overset approaches were not developed for finite element methods until 2010's (Massing et al., 2013). Galbraith et al. developed the first high-order overset DG scheme (Galbraith et al., 2015). In their method the domain was decomposed into multiple overlapping structured meshes, where each mesh can utilize the approximation order independently. Brazell et al. further extended this overset mesh method (Brazell et al., 2016) by using an overset framework called Topology Independent Overset

Grid Assembler (TIOGA) (Roget & Sitaraman, 2014). In the TIOGA framework, unstructured and mix-element meshes are allowed.

For any overset method, several core functionalities need to be established, i.e. hole-cutting, receptor-donor connectivity, and data communication. Hole cutting removes elements in the background mesh which overlap the near-body mesh. The hole boundaries need solution data interpolated from the near-body mesh, while the outer-boundary of the near-body mesh gets data from the background mesh. The setup of the receptor-donor connectivity then establishes a "bridge" at the overset interfaces for the data communications. Most overset approaches update the flow field at each mesh separately. The field data is interpolated at the donor cells and then sent to the receptor points of the other mesh. Data received at the receptor points is often used as boundary conditions imposed on the overset interfaces. This approach is known as artificial boundary (AB) approach (Galbraith, 2013). In some cases the core functionalities are overlapped or combined. For example the hole cutting procedure is synchronous with the receptor-donor connectivity, or the build of receptor-donor connectivity directly yields interpolation weights.

## 1.4 Extension to Moving Grids

The first CFD simulation of rotorcraft using overset moving grids can be traced back to 90's (MEAKIN, 1993). In this work the complete V-22 tiltrotor aircraft geometry, including the rotor blades and the fuselage was simulated. Since then several developments have been applied for helicopter applications (Ahmad & Duque, 1996; Stangl & Wagner, 1996). Overset approaches were also extended to other applications such as casing treatment in turbomachinery (Legras et al., 2009) and flow-structure interactions (Miller et al., 2014). More recently, high-order overset FR/CPR methods were developed for moving grids (Crabill et al., 2016a; Duan & Wang, 2019) and applied to the simulation of a rotor blade. Since the meshes have relative motions, the hole cutting and receptor-donor connectivity need to be updated at each time step. Special care needs to be taken for cells that are cutoff at the current time step and brought back at the next time step. We will address this in more details in Chapter 3.

Another technique for moving boundary problems is the sliding mesh technique. A sliding surface can be defined as the boundary between two non-conformed meshes. The non-conformed meshes were first used as stationary meshes to simplify mesh generation for structured meshes or complex geometries (Rai, 1986; Fillola et al., 2004). They were then used as sliding meshes for simulations with bodies that have relative motions, such as flow in stirred reactors (Bakker et al., 1997), a helicopter rotor and fuselage (Steijl & Barakos, 2008), and a tidal-stream turbine (McNaughton et al., 2014). In these simulations, prescribed motions were defined on the sliding surface and imposed on each of the mesh independently. The sliding mesh technique was also incorporated into high-order methods for flow problems with rotational bodies. Ferrer et al. developed a DG incompressible flow solver for sliding meshes (Ferrer & Willden, 2012). Liang et al. developed a spectral difference method (Zhang & Liang, 2015a; Zhang et al., 2016) and FR/CPR method (Zhang & Liang, 2015b) for solving 2D compressible flow on meshes with rotational motions and deformations.

Sliding mesh can be viewed as a special case of overset meshes. They both involve multiple moving grids, have receptor-donor connectivity updated at each time step, and communicate interpolated flow data between meshes. Unlike overset meshes, the sliding meshes do not overlap but only contact each other. Therefore no hole cutting is needed for sliding meshes. However, the relative motions between meshes have to be constrained on the sliding interface, which makes the sliding mesh technique not as general as the moving overset mesh technique. Typically, an arbitrary motion is allowed in overset methods as long as the meshes overlap each other. The comparison of simulation results from overset meshes and sliding meshes for a flow over counter rotating open-rotors case can be found in (Francois et al., 2011).

## 1.5 Overview of the Dissertation

Chapter 2 introduces the high-order FR/CPR method. The basic idea in 1D and its formulations for 2D/3D are presented. The implementation of the FR/CPR method on grids with curved elements is also presented. Several methods for computing the common viscous fluxes and their derivatives

are briefly introduced.

In Chapter 3 the overset method for this study is presented with details. An efficient parallel hole cutting method is described. This method uses Cartesian grids as the approximate geometry representations and does query-based inside/outside checking to mark the cells to be cut out. The establishment of receptor-donor connectivity is performed after hole cutting. Two different data communication approaches are then compared. The extension to moving grids and sliding meshes is described at the end of the chapter.

Chapter 4 presents the time integration methods using for the overset method. Two and three-stage Runge-Kutta methods are briefly introduced. These methods are chosen as explicit time marching schemes for the flow solver. For the implicit schemes, the backward Euler method is used as the first order scheme and the Crank-Nicolson method is for the second order scheme. These implicit schemes are combined with the block lower-upper symmetric Gauss-Seidel (LU-SGS) method to solve the non-linear equations at each time step.

In Chapter 5 several simulation cases are demonstrated to verify and validate the high-order FR/CPR overset method for both stationary and moving grids. The stability of different overset data communication approaches are compared using a vortex propagation case. Then accuracy studies are performed on stationary overset, moving overset, and sliding meshes. The convecting vortex problem is used for inviscid flows and the Couette problem is used for viscous flows. Low Reynolds number flow over a sphere is then simulated. The convergent steady flow fields are obtained at different orders of accuracy. To demonstrate that the high-order FR/CPR overset method is capable of performing implicit large eddy simulation, a transitional flow over the T106A turbine blade case is run on stationary overset meshes and a single-domain mesh for comparison. Another case for T106A blade put the blade in the wake region of moving cylinders. In this case the cylinder mesh and the blade mesh contact each other at a common interface. The cylinder mesh slides along the interface while the blade mesh remains stationary. The pressure coefficient and skin friction coefficient distributions under the influence of upwind cylinders are compared with those without the cylinders. To further verify the moving overset solver, a pitching NACA0012 airfoil

case from the 4th International Workshop on High-Order CFD Methods (https://how4.cenaero.be) was simulated. The results are then compared with available data from the Workshop. The final case in this Chapter is a single-bladed rotor hovering problem. The tip vortices are resolved and the features of these vortices such as the peak swirl velocity, the vortex core radius, and the position of the vortex center, are compared with experimental results.

Finally, Chapter 6 concludes with a summary of the work presented in this dissertation, as well as thoughts about future work.

# Chapter 2

# High-Order FR/CPR Method

This chapter reviews the high-order FR/CPR method. It starts with the basic idea and the formulations in 1D. Various choices of the correction function are discussed. Then the extension to multidimensional and high-order elements are presented. Several methods for computing the common viscous fluxes and their derivatives are introduced. The extension to dynamic moving grids is also discussed.

## 2.1 Framework of the FR/CPR Method

It is convenient to start the derivations in one dimensional space. Consider the conservation law

$$\frac{du}{dt} + \frac{df}{dx} = 0, \tag{2.1}$$

with initial condition $u(x,0) = u_0(x)$ and the flux $f$ depends on $u$. The initial solution $u_0$ is periodic or of compact support so that boundary conditions are trivial. Let $a(u) \equiv df/du$. Then the above equation can be cast in a non-conservative form

$$\frac{du}{dt} + a(u)\frac{du}{dx} = 0. \tag{2.2}$$

Let the domain (a line in this case) be divided into a set of cells or elements $\{E_j\}$, $j = 1, 2, \cdots$. On each cell, let the solution be approximated by a degree $K-1$ polynomial built from solutions $u_{j,k}$ at $K$ solution points $x_{j,k}$. The $K$ solution points are typically the Gauss points or Lobatto points. For convenience, the type of solution points is assumed to be the same for all cells. Note that if

Lobatto points are selected, each interface $x_{j+1/2}$ has two solution values of $u$, namely $u_{j,K}$ and $u_{j+1,1}$. These left and right values are readily available for the calculation of the common flux. As a standard in finite element methods, it is more convenient to map each element to a single reference element, i.e. the interval $I = [-1,1]$, instead of dealing with global elements $\{E_j\}$. $E_j$ is often regarded as the physical domain and $I$ as the computational domain. Denote the center of $E_j$ by $x_j$ and its width by $h_j$. With the reference coordinate $\xi$ defined on $I$, the mapping I onto $E_j$ and its inverse are

$$\begin{cases} x(\xi) = x_j + \xi h_j/2 \\ \xi(x) = 2(x - x_j)/h_j \end{cases}. \tag{2.3}$$

The reference coordinates of solution points are denoted by $\xi_k, k = 1 \cdots K$. The derivative of a function $r_j(x)$ defined on $E_j$ can be computed by the chain rule

$$\frac{dr_j}{dx} = \frac{2}{h_j}\frac{dr_j(\xi)}{d\xi}, \tag{2.4}$$

where $r_j(\xi) = r_j(x(\xi))$. With these definitions and settings we can start to go into the basics of the 1D FR/CPR method.

### 2.1.1 Basic Formulations in 1D Space

When one solves the conservation law Eq. (2.1), the first step is to approximate $u$ on each cell by a polynomial of degree $K-1$. Lagrange polynomials are often chosen as basis functions to construct the solution polynomials $u_j(x)$ or $u_j(\xi)$. The Lagrange polynomials defined at solution point $k$ are

$$\phi_{j,k}(x) = \prod_{l=1, l\neq k}^{K} \frac{x - x_{j,l}}{x_{j,k} - x_{j,l}} \quad \text{and} \quad \phi_k(\xi) = \prod_{l=1, l\neq k}^{K} \frac{\xi - \xi_l}{\xi_k - \xi_l}. \tag{2.5}$$

The solution polynomials can then be computed as the linear combination of basis functions:

$$u_j(x) = \sum_{k=1}^{K} u_{j,k}\phi_{j,k}(x) \quad \text{and} \quad u_j(\xi) = \sum_{k=1}^{K} u_{j,k}\phi_k(\xi). \tag{2.6}$$

Similarly, the flux polynomials are constructed as

$$f_j(x) = \sum_{k=1}^{K} f_{j,k}\phi_{j,k}(x) \quad \text{and} \quad f_j(\xi) = \sum_{k=1}^{K} f_{j,k}\phi_k(\xi). \tag{2.7}$$

The conservation law Eq. (2.1) requires to compute their derivative at solution points. With above representations of the flux functions, the derivatives at solution point $l$ can be calculated by

$$\left.\frac{df_j(\xi)}{d\xi}\right|_{\xi=\xi_l} = \sum_{k=1}^{K} f_{j,k}\left.\frac{d\phi_k(\xi)}{d\xi}\right|_{\xi=\xi_l} = \sum_{k=1}^{K} f_{j,k}d_{l,k}, \tag{2.8}$$

and

$$\left.\frac{df_j(x)}{dx}\right|_{x=x_l} = \frac{2}{h_j}\left.\frac{df_j(\xi)}{d\xi}\right|_{\xi=\xi_l}. \tag{2.9}$$

An alternative way to calculate the derivatives is the chain rule

$$\left.\frac{df_j(\xi)}{d\xi}\right|_{\xi=\xi_l} = \left.\frac{df_j}{du}\frac{du}{d\xi}\right|_{\xi=\xi_l} = a(u_{j,l})\left.\frac{du}{d\xi}\right|_{\xi=\xi_l} \tag{2.10}$$

We now define the various left and right values at the cell interfaces. Let subscript L denote the value at left side of the interface and subscript R denote the value at the right side. The solution and flux values are given by

$$\begin{cases} u_L = u_j(x_{j+1/2}) \\ u_R = u_{j+1}(x_{j+1/2}) \end{cases}, \quad \text{and} \quad \begin{cases} f_L = f_j(x_{j+1/2}) \\ f_R = f_{j+1}(x_{j+1/2}) \end{cases}. \tag{2.11}$$

In general, $u_L \neq u_R$, $f_L \neq f_R$, $f_L \neq f(u_L)$, and $f_R \neq f(u_R)$. Therefore the flux functions are often called discontinuous flux functions. To construct a continuous flux function, an upwind flux at the cell interface is used. This flux is named common flux, which corresponds to the numerical flux

for the DG methods. At $x_{j+1/2}$, let $\tilde{u}$ be defined by the mean value theorem

$$a(\tilde{u}) = \begin{cases} (f(u_R) - f(u_L))/(u_R - u_L) & \text{if} \quad u_L \neq u_R, \\ a(u_L) = a(u_R) & \text{otherwise.} \end{cases} \tag{2.12}$$

Then the common flux is determined by the sign of $a(\tilde{u})$,

$$f_{com} = f_{upwind} = \begin{cases} f(u_L) & \text{if} \quad a(\tilde{u}) \geq 0, \\ f(u_R) & \text{otherwise.} \end{cases} \tag{2.13}$$

An alternative for the common flux is

$$f_{com} = \frac{1}{2}[f(u_L) + f(u_R)] - \frac{1}{2}|a(\tilde{u})|(u_R - u_L). \tag{2.14}$$

To construct the continuous flux function $F_j$ on cell $j$, several conditions need to be satisfied: 1) take the common flux values at the cell interfaces, i.e. $F_j(x_{j-1/2}) = f_{j-1/2,com}$ and $F_j(x_{j+1/2}) = f_{j+1/2,com}$; 2) be close to $f_j$; and 3) be a polynomial of degree $K$. The first two requirements are straightforward. The reason for third requirement is to keep the order of accuracy when updating the solution $u_j$. In the conservation law Eq. (2.1) the derivative on $F_j$ decrease the degree by one. If $F_j$ is of degree $K$, then $\frac{dF_j}{dx}$ is of degree $K-1$, which is of the same degree with $u_j$. The increment $\Delta t \frac{dF_j}{dx}$ has the same degree with the solution polynomial will guarantee the updated solution also to be of the same degree.

In practice, instead of explicitly constructing $F_j$, the difference $F_j - f_j$ is defined, which approximates the zero function. This difference is named the correction. In the computational domain, the corrections at the cell interfaces are

$$\begin{cases} F_j(-1) - f_j(-1) = f_{j-1/2,com} - f_j(-1), \\ F_j(1) - f_j(1) = f_{j+1/2,com} - f_j(1). \end{cases} \tag{2.15}$$

Therefore, $F_j - f_j$ takes on above prescribed left and right correction values, is of degree $K$, and approximate the zero function. One more requirement is that the correction at the left interface is separated from that of the right interface. As shown later, this separation guarantees the conservative of the FR/CPR method. For this reason two correction functions, $g_{LB}(\xi)$ and $g_{RB}(\xi)$, are defined. 'LB' stands for the left boundary and 'RB' stands for the right boundary. They are both of degree $K$. The separation is satisfied by the conditions

$$g_{LB}(-1) = 1,\ g_{LB}(1) = 0; \quad g_{RB}(-1) = 0,\ g_{RB}(1) = 1 \tag{2.16}$$

For simplicity reason, the two functions are chosen to reflect each other, i.e. $g_{RB}(\xi) = g_{LB}(-\xi)$. Thus $F_j$ can be constructed as

$$F_j(\xi) = f_j(\xi) + [f_{j-1/2,com} - f_j(-1)]g_{LB}(\xi) + [f_{j+1/2,com} - f_j(1)]g_{RB}(\xi). \tag{2.17}$$

Finally, the derivative of $F_j(\xi)$ at solution point $\xi_k$ is

$$F_j'(\xi_k) = f_j'(\xi_k) + [f_{j-1/2,com} - f_j(-1)]g_{LB}'(\xi) + [f_{j+1/2,com} - f_j(1)]g_{RB}'(\xi). \tag{2.18}$$

In the physical domain, the derivative is

$$\left.\frac{dF_j}{dx}\right|_{x=x_k} = \frac{2}{h_j}F_j'(\xi_k). \tag{2.19}$$

The resulting scheme for the conservation law at $x_{j,k}$ is

$$\frac{du_{j,k}}{dt} = -\left.\frac{dF_j}{dx}\right|_{x=x_k}. \tag{2.20}$$

We now show that this scheme is conservative. The integral form of Eq. (2.1) is

$$\frac{\partial}{\partial t}\int_{E_j} u(x,t)dx = -\int_{E_j}\frac{df}{dx} = f_{j-1/2,com} - f_{j+1/2,com}. \tag{2.21}$$

Suppose existing a quadrature rule that is exact for any polynomial of degree $K-1$ or less. The weight at $\xi_k$ is $w_k$. Then

$$\int_{-1}^{1} u_j(\xi)d\xi = \sum_{k=1}^{K} w_k u_{j,k}. \tag{2.22}$$

Switching to physical domain,

$$\int_{E_j} u_j(x)dx = \frac{h_j}{2}\int_{-1}^{1} u_j(\xi)d\xi = \frac{h_j}{2}\sum_{k=1}^{K} w_k u_{j,k}. \tag{2.23}$$

The left hand side of Eq. (2.21) can be written as

$$\frac{\partial}{\partial t}\int_{E_j} u(x,t)dx = \frac{h_j}{2}\frac{\partial}{\partial t}\sum_{k=1}^{K} w_k u_{j,k} = \frac{h_j}{2}\sum_{k=1}^{K} w_k \frac{du_{j,k}}{dt} = -\frac{h_j}{2}\sum_{k=1}^{K} w_k \frac{dF_j}{dx}\bigg|_{x=x_k}. \tag{2.24}$$

Since $\frac{dF_j}{dx}$ is of degree $K-1$, the right most term of above equality can be transformed as

$$-\frac{h_j}{2}\sum_{k=1}^{K} w_k \frac{dF_j}{dx}\bigg|_{x=x_k} = -\int_{E_j}\frac{dF_j}{dx}dx. \tag{2.25}$$

From the fundamental theorem of calculus,

$$-\int_{E_j}\frac{dF_j}{dx}dx = F_j(x_{j-1/2}) - F_j(x_{j+1/2}) = f_{j-1/2,com} - f_{j+1/2,com}. \tag{2.26}$$

The above three equations imply 2.21, which means the scheme is conservative.

### 2.1.2 Correction Functions

Next, we need to determine the correction functions of degree $K$ polynomials. Since $g_{RB}(\xi) = g_{LB}(-\xi)$, we only focus on $g_{LB}(\xi)$ and then reflect it to $g_{RB}(\xi)$. Various special polynomials are available for the correction functions. Before jumping into those polynomials we need to review the mathematics related to the linear space of polynomials. Let $\mathbf{P}_m$ denote the linear space of polynomials of degree $m$ or less defined on the range of $I = [-1,1]$. The inner product of any two

polynomials $v, w \in \mathbf{P}_m$ is defined by

$$(v,w) = \int_{-1}^{1} v(\xi) w(\xi) d\xi. \tag{2.27}$$

A polynomial $v$ is said to be orthogonal to $\mathbf{P}_m$ if for each $l$, $0 \leq l \leq m$,

$$(v, \xi^l) = \int_{-1}^{1} v(\xi) \xi^l d\xi = 0. \tag{2.28}$$

The first polynomials we look at are the Legendre polynomials, which are given by a recurrence formula (Hildebrand, 1987),

$$\begin{gathered} P_0 = 1, \quad P_1 = \xi, \\ P_k(\xi) = \frac{2k-1}{k} \xi P_{k-1}(\xi) + \frac{k-1}{k} P_{k-2}(\xi), \quad \text{for } k > 2, \end{gathered} \tag{2.29}$$

where $k$ is the degree of the polynomials. The Legendre polynomials have the following properties:

$$\begin{gathered} P_k(-1) = (-1)^k, \qquad P_k(1) = 1, \\ (P_k, P_l) = 0 \text{ for } k \neq l, \quad (P_k, P_k) = \frac{2}{2k+1}. \end{gathered} \tag{2.30}$$

The zeros of $P_k$ are the $k$ Gauss points on $[-1,1]$. The derivatives at the end points are

$$\begin{cases} P_k'(-1) = \dfrac{(-1)^{k-1} k(k+1)}{2}, \\ P_k'(1) = \dfrac{k(k+1)}{2}. \end{cases} \tag{2.31}$$

The next polynomial is the right Radau polynomial. The degree $k$ Radau polynomial is given by

$$R_{R,k} = \frac{(-1)^k}{2} (P_k - P_{k-1}). \tag{2.32}$$

The properties of the Radau polynomial are

$$R_{R,k}(-1) = 1, \quad R_{R,k}(1) = 0, \tag{2.33}$$

and $R_{R,k}$ is orthogonal to $\mathbf{P}_{k-2}$, which means it approximates the zero function in the sense of least squares. These properties make the Radau polynomial a natural choice for the correction function. The zeros of the Radau polynomial are the right Radau points, and the derivative at the end points are

$$\begin{cases} R'_{R,k}(-1) = \dfrac{-k^2}{2}, \\ R'_{R,k}(1) = \dfrac{(-1)^{k-1}k}{2}. \end{cases} \tag{2.34}$$

The third polynomial is the Lobatto polynomial. The degree $k$ Lobatto polynomial is defined by

$$Lo_k = P_k - P_{k-2}, \tag{2.35}$$

or

$$Lo_k = 2(-1)^k(R_{R,k} - R_{R,k-1}). \tag{2.36}$$

The zeros of the Lobatto polynomial are the $k$ Lobatto points including the two boundaries $\xi = \pm 1$, which gives

$$Lo_k(-1) = Lo_k(1) = 0. \tag{2.37}$$

The degree $k$ Lobatto polynomial is orthogonal to $\mathbf{P}_{k-3}$. The derivatives at the end points are

$$\begin{cases} Lo'_k(-1) = (-1)^{k+1}(2k-1), \\ Lo'_k(1) = -2k+1. \end{cases} \tag{2.38}$$

Since we only focus on $g_{LB}$ for now, for simplicity of notation we set $g \equiv g_{LB}$. Since $g$ is of degree $K$, it is completely determined by $K+1$ conditions. Two conditions are known, namely,

$$g(-1) = 1, \quad g(1) = 0. \tag{2.39}$$

There are $K-1$ conditions remain. Due to the properties of the Radau polynomial in Eq. (2.33), which satisfies the above two conditions, a generally expression satisfying above two conditions for the correction function is written as

$$g = \alpha R_{R,K} + (1-\alpha) R_{R,K-1}, \tag{2.40}$$

where $\alpha$ is a coefficient satisfies $0 \leq \alpha \leq 1$ and needs to be determined. Different choices of $g$ lead to different schemes including DG, spectral difference, spectral volume and so on. Three choices are reviewed here. The first choice for $g$ is to let $\alpha = 1$, denoted by $g = g_1 = R_{R,K}$. It uses the property of $R_{R,K}$, which is orthogonal to $\mathbf{P}_{K-2}$, to give the $K-1$ conditions. The resulting scheme is identical to DG. Therefore, $g_1$ is also denoted by $g_{DG}$.

The second choice for $g$, denoted by $g_2$, is defined by $\alpha = \frac{K-1}{2K-1}$, i.e.

$$g_2 = \frac{K-1}{2K-1} R_{R,K} + \frac{K}{2K-1} R_{R,K-1}. \tag{2.41}$$

This choice make $g_2$ orthogonal to $\mathbf{P}_{K-3}$, which gives $K-2$ conditions. The last condition is $g_2'(1) = 0$, which can be verified by Eq. (2.34). $g_2$ has the remarkable property that $g_2'$ vanishes at the $K$ Lobatto points except for the left boundary $\xi = -1$. Therefore, it is convenient and economical to select the $K$ Lobatto points as solution points for $g_2$. With such a selection, the flux jump at the interface results in a correction to only $\frac{df_j}{d\xi}|_{\xi=\xi_1}$ but not to any $\frac{df_j}{d\xi}|_{\xi=\xi_k}, k > 1$. That is, the correction at the left boundary is lumped into that boundary. For this reason, $g_2$ is also denoted by $g_{Lump,Lo}$. Employing Eq. (2.34) with $k = K$ and then $k = K-1$, the correction at the

left boundary is calculated by

$$g_2'(-1) = -\frac{K(K-1)}{2}. \tag{2.42}$$

The third choice for $g$, denoted by $g_{Ga}$, requires that $g$ vanishes at the $K-1$ Gauss points, which gives $K-1$ conditions. These points are the zeros of the Legendre polynomial $P_{K-1}$. It can be verified that

$$g_{Ga} = \frac{K}{2K-1}R_{R,K} + \frac{K-1}{2K-1}R_{R,K-1}, \tag{2.43}$$

which implies $\alpha = \frac{K}{2K-1}$.

Fourier analysis shows that if $g$ is orthogonal to $\mathbf{P}_m$, the resulting scheme is accurate and stable to order $K+m+1$. Therefore, the scheme using $g_{DG}$ is stable and accurate to order $2K-1$, and schemes using $g_2$ and $g_{Ga}$ are stable and accurate to order $2K-2$. On the other hand, $g_{DG}$ is the steepest among the three correction function while $g_2$ is the least steep. A steeper correction function tends to result a scheme with smaller CFL limit. In fact, the CFL limit resulting from $g_2$ is roughly twice as large as that from $g_{DG}$. As the conclusion, the schemes in the order of decreasing accuracy as well as increasing time-stepping limit are: $g_{DG}$, $g_{Ga}$, and $g_{Lump,Lo}$. For the Fourier analysis and more details of the FR/CPR method, see (Huynh, 2007, 2009).

## 2.2 FR/CPR Method in Multi-Dimensional Space

We first start with the Euler equations for inviscid flow in 2D space

$$\frac{\partial Q}{\partial t} + \nabla \cdot \vec{F}(Q) = 0, \tag{2.44}$$

where $Q = [\rho, \rho u, \rho v, \rho E]^T$ are the conservative variables, $\rho$ is the density, $u, v$ are the velocity components in each spatial coordinate direction, $E$ is the specific total energy, and

$$\vec{F}(Q) = (F, G) = \left( \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{bmatrix}, \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \end{bmatrix} \right) \tag{2.45}$$

are the inviscid fluxes, where $p = (\gamma - 1)(\rho E - \frac{\rho}{2}(u^2 + v^2))$ is the pressure for an ideal perfect gas, $H = E + p/\rho$ is the specific total enthalpy. For simplicity we restrict the derivations in 2D space. The formulas in 3D can be derived in the same manner. Assume that the domain $\Omega$ is discretized into $N$ non-overlapping elements $\{V_i\}_{i=1}^N$. Let $W$ be an arbitrary weighting function or test function. Similar to Galerkin method, the weighted residual integral form of Eq. (2.44) on element $V_i$ can be written as

$$\int_{V_i} (\frac{\partial Q}{\partial t} + \nabla \cdot \vec{F}(Q)) W dV = 0. \tag{2.46}$$

Performing integration by parts to the flux divergence, we obtain

$$\int_{V_i} \frac{\partial Q}{\partial t} W dV + \int_{\partial V_i} W \vec{F}(Q) \cdot \vec{n} dS - \int_{V_i} \nabla W \cdot \vec{F}(Q) dV = 0. \tag{2.47}$$

Let $\mathbf{P}_k(V_i)$ denote the space of polynomial of degree $k$ defined on $V_i$. Let $Q_i \in \mathbf{P}_k(V_i)$ be an approximate solution to the analytical solution $Q$. The approximate flux $\vec{F}(Q_i)$ is generally discontinuous across element interfaces. To achieve conservation, a continuous flux function needs to be constructed. Following the idea used in the Godnov method (Roe, 1981; Rusanov, 1962), the normal flux term in Eq. (2.47) is replaced with a common Riemann flux

$$F^n(Q_i) \equiv \vec{F}(Q_i) \cdot \vec{n} \approx F^n_{com}(Q_i^-, Q_i^+, \vec{n}), \tag{2.48}$$

where the $-,+$ superscripts indicate the trace of the conservative variables on the element's boundary interior and exterior respectively. Replacing the flux at the element's boundary with the common flux, we obtain

$$\int_{V_i} \frac{\partial Q_i}{\partial t} W dV + \int_{\partial V_i} W F^n_{comm} dS - \int_{V_i} \nabla W \cdot \vec{F}(Q_i) dV = 0. \tag{2.49}$$

Applying integration by parts again to the third term of the above equation, we get

$$\int_{V_i} \frac{\partial Q_i}{\partial t} W dV + \int_{V_i} W \nabla \cdot \vec{F}(Q_i) dV + \int_{\partial V_i} W [F^n_{com} - F^n(Q_i)] dS = 0. \tag{2.50}$$

The test space here has the same dimension as the solution space, and is chosen in a manner to guarantee the existence and uniqueness of the numerical solution. Note that the divergence of the flux $\nabla \cdot \vec{F}(Q_i)$ involves no influence from the data from the neighboring cells. The influence of these data is represented by above boundary integral.

The next step is to eliminate the test function, the boundary integral in the above equation is cast as a volume integral by introducing a "correction field" $\delta_i \in \mathbf{P}_k(V_i)$,

$$\int_{V_i} W \delta_i d\Omega \equiv \int_{\partial V_i} W [F^n] dS, \tag{2.51}$$

where $[F^n] \equiv F^n_{com} - F^n(Q_i)$ is the normal flux difference on the boundary of the element. The above equation is sometimes called the "lifting operator", which has the normal flux difference on the boundary as input and an element of $\mathbf{P}_k(V_i)$ as output. Substitute Eq. (2.51) into Eq. (2.50) we get

$$\int_{V_i} (\frac{\partial Q_i}{\partial t} + \nabla \cdot \vec{F}(Q_i) + \delta_i) W dV = 0. \tag{2.52}$$

If the flux vector $\vec{F}(Q_i)$ is a linear function of $Q_i$, then $\nabla \cdot \vec{F}(Q_i) \in \mathbf{P}_k(V_i)$. In this case, the terms inside the parentheses of above equation are all elements of $\mathbf{P}_k(V_i)$. Because the test space is

selected to ensure a unique solution, Eq. (2.52) is equivalent to

$$\frac{\partial Q_i}{\partial t} + \nabla \cdot \vec{F}(Q_i) + \delta_i = 0. \tag{2.53}$$

For non-linear conservation laws, the $\vec{F}(Q_i)$ is not a linear function of $Q_i$. As a result, $\nabla \cdot \vec{F}(Q_i)$ is usually not an element of $\mathbf{P}_k(V_i)$. Eq. (2.52) cannot be reduced to Eq. (2.53). However, if $\nabla \cdot \vec{F}(Q_i)$ is projected to $\mathbf{P}_k(V_i)$ by

$$\int_{V_i} \Pi(\nabla \cdot \vec{F}(Q_i)) W dV = \int_{V_i} \nabla \cdot \vec{F}(Q_i) W dV, \tag{2.54}$$

where $\Pi$ is a projection operator, then Eq. (2.52) reduces to a differential form

$$\frac{\partial Q_i}{\partial t} + \Pi(\nabla \cdot \vec{F}(Q_i)) + \delta_i = 0. \tag{2.55}$$

To this point the weighted residual formulation is reduced to a differential formulation. Note that for $\delta_i$ defined by Eq. (2.51), if $W \in \mathbf{P}_k$, Eq. (2.55) is equivalent to the DG formulation. If $W$ belongs to other spaces, the resulting $\delta_i$ is different, which leads to a different method such as spectral volume method.

In the FR/CPR method, the approximate solution $Q_i$ is represented by a degree $k$ polynomial constructed from the solution values at the solution points (SPs) $\{\vec{r}_{ij}\}, j = 1, \cdots K$. Here $K$ is the dimension of $\mathbf{P}_k(V_i)$. For a triangle $K = (k+1)(k+2)/2$, and for a quadrilateral $K = (k+1)^2$. Lagrange polynomials defined inside element $i$ can be used to construct $Q_i$:

$$Q_i(\vec{r}) = \sum_j L_j(\vec{r}) Q_{i,j}, \tag{2.56}$$

where $j$ is the index of the solution points, $L_j(\vec{r})$ is the Lagrange interpolation polynomial defined

on SP $j$, and $Q_{i,j}$ is the solution value at SP $j$. Then Eq. (2.55) holds true at SPs, i.e.

$$\frac{\partial Q_{i,j}}{\partial t} + \Pi_j(\nabla \cdot \vec{F}(Q_i)) + \delta_{i,j} = 0, \tag{2.57}$$

where $\Pi_j(\nabla \cdot \vec{F}(Q_i))$ denotes the values of $\Pi(\nabla \cdot \vec{F}(Q_i))$ at SP $j$. To compute $\delta_i$, $k+1$ points are defined on each interface of element $i$. These points are named flux points (FPs). The value of the normal flux difference $[F^n]$ are computed and stored at the FPs. $[F^n]$ is then approximated with a degree $k$ polynomial along each interface,

$$[F^n]_f \approx \mathbf{I}_k[F^n]_f \equiv \sum_l [F^n]_{f,l} L_l^{FP}, \tag{2.58}$$

where $f$ is the index of the interface, $l$ is the FP index, and $L_l^{FP}$ is the Lagrange interpolation polynomial based on the FPs in a local interface coordinate. An example of solution points and flux points of a triangle is shown in Figure 2.1. For a linear element with straight edges, once the solution points and the flux points are chosen, correction field at the SPs is computed as

$$\delta_{i,j} = \frac{1}{|V_i|} \sum_{f \in \partial V_i} \sum_l \alpha_{j,f,l} [F^n]_{f,l} S_f, \tag{2.59}$$

where $\alpha_{j,f,l}$ are the lifting coefficients which are independent of the solutions, $S_f$ is the area of face $f$, and $|V_i|$ is the volume of $V_i$. Therefore, the final form can be written as

$$\frac{\partial Q_{i,j}}{\partial t} + \Pi_j(\nabla \cdot \vec{F}(Q_i)) + \frac{1}{|V_i|} \sum_{f \in \partial V_i} \sum_l \alpha_{j,f,l} [F^n]_{f,l} S_f = 0. \tag{2.60}$$

The term $\Pi_j(\nabla \cdot \vec{F}(Q_i))$ can be directly computed from Eq. (2.54) once the test function $W$ is given. However, it is computationally expensive since high-order integral quadratures are required. Two efficient approaches, namely Lagrange polynomial approach (LP) and chain rule approach (CR), are developed to approximate the projection. In the LP approach the flux vector is approximated

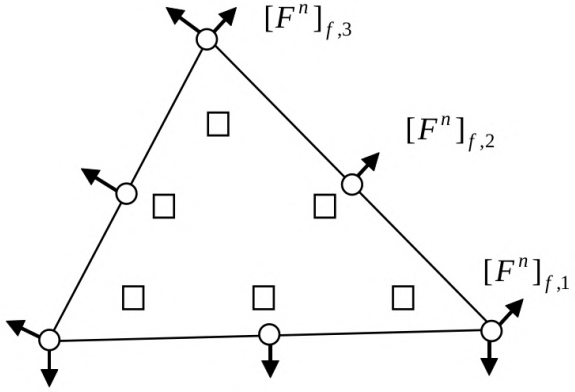



Figure 2.1: An Example of solution points (squares) and flux points (circles) for a triangle.

by degree $k$ Lagrange interpolation polynomials:

$$\vec{F}(Q_i) \approx \sum_j L_j^{SP}(\vec{r})\vec{F}(Q_{i,j}), \tag{2.61}$$

where $L_j^{SP}(\vec{r})$ is the Lagrange polynomial based on the solution point at $r_{i,j}$. The projection is computed with the divergence operating on the Lagrange polynomials, i.e.

$$\Pi(\nabla \cdot \vec{F}(Q_i)) \approx \sum_j \nabla L_j^{SP}(\vec{r}) \cdot \vec{F}(Q_{i,j}). \tag{2.62}$$

In this approach, $\Pi(\nabla \cdot \vec{F}(Q_i))$ is a degree $k-1$ polynomial which also belongs to $\mathbf{P}_k$. Numerical experiments indicates that there is a slight loss of accuracy with the LP approach, but it is fully conservative.

In the CR approach, the divergence of the flux vector $\vec{F} = (F, G)$ is computed analytically,

$$\begin{aligned} \nabla \cdot \vec{F}(Q_{i,j}) &= \frac{\partial F(Q_{ij})}{\partial x} + \frac{\partial G(Q_{ij})}{\partial y} \\ &= \frac{\partial F(Q_{ij})}{\partial Q}\frac{\partial Q_{i,j}}{\partial x} + \frac{\partial G(Q_{ij})}{\partial Q}\frac{\partial Q_{i,j}}{\partial y} \\ &= \frac{\partial \vec{F}(Q_{i,j})}{\partial Q} \cdot \nabla Q_{i,j}, \end{aligned} \tag{2.63}$$

where $\frac{\partial \vec{F}(Q_{i,j})}{\partial Q}$ is the flux Jacobian matrix, which can be computed analytically. The projection is

the approximated by the Lagrange interpolation polynomials

$$\Pi(\nabla \cdot \vec{F}(Q_i)) \approx \sum_j L_j^{SP}(\vec{r}) \nabla \cdot \vec{F}(Q_{i,j}). \tag{2.64}$$

Numerical experiments show that the CR approach is more accurate than the LP approach, at the expense of full conservation. For more details of the two approaches, see (Wang & Gao, 2009).

## 2.3 Extension to High-Order Elements

Since high-order elements with curved edges represent the geometry better than linear elements, it is preferred to use high-order elements for high-order methods. In this section, we present the extension of FR/CPR to high-order elements. General triangular and quadrilateral elements with possible high-order edges are considered. To achieve an efficient implementation, all elements are transformed from the physical domain to a standard element in the computational domain represented by the coordinates $(\xi, \eta)$. The standard triangle is

$$\mathbf{T} = \left\{ \vec{\xi} = (\xi, \eta) \middle| (\xi, \eta) \geq 0; \xi + \eta \leq 1 \right\}, \tag{2.65}$$

and the standard quadrilateral is

$$\mathbf{Q} = \left\{ \vec{\xi} = (\xi, \eta) \middle| -1 \leq \xi \leq 1; -1 \leq \eta \leq 1 \right\}. \tag{2.66}$$

The transformation is shown in Figure 2.2. The transformation can be written as

$$\vec{r} = \sum_j M_j(\vec{\xi}) \vec{r}_j, \tag{2.67}$$

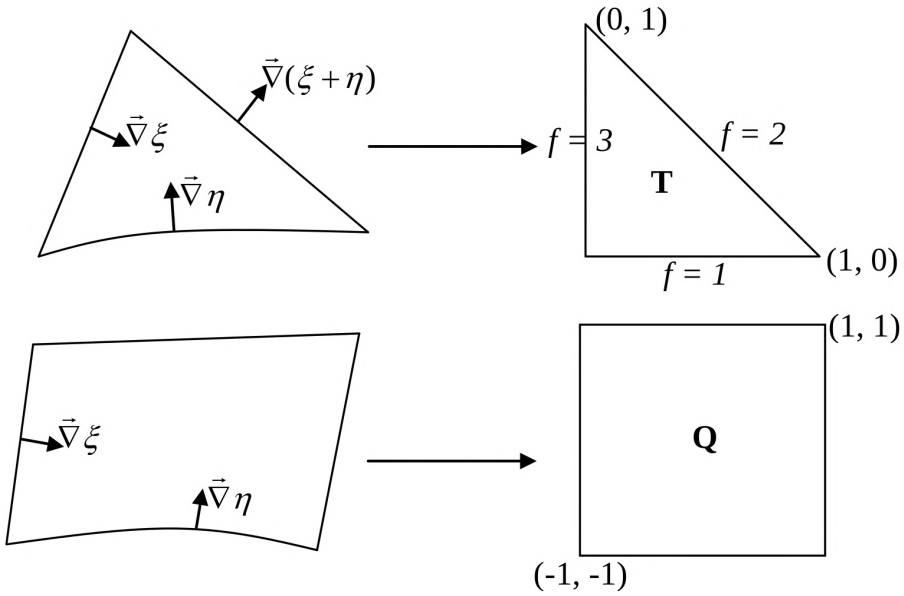



Figure 2.2: Transformation of general elements to standard elements.

where $j$ is the index of nodes of the element, $\vec{r}_j$ is the physical coordinates of node $j$, and $M_j(\vec{\xi})$ is the shape functions. The Jacobian matrix is computed as

$$J = \frac{\partial \vec{r}}{\partial \vec{\xi}} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}, \tag{2.68}$$

where the subscripts denote taking derivative with respect to that variable. The inverse of the Jacobian matrix can be computed by

$$J^{-1} = \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = \frac{1}{|J|} \begin{bmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{bmatrix}. \tag{2.69}$$

The Euler equation (2.44) is then transformed into the following form

$$\frac{\partial \tilde{Q}}{\partial t} + \nabla^\xi \cdot \vec{\tilde{F}} = 0, \tag{2.70}$$

where $\vec{\tilde{F}} = (\tilde{F}, \tilde{G})$, $\nabla^{\xi}$ is the divergence operator in the computational domain, and

$$\begin{cases} \tilde{Q} = |J|Q, \\ \tilde{F} = |J|(\xi_x F + \xi_y G), \\ \tilde{G} = |J|(\eta_x F + \eta_y G). \end{cases} \tag{2.71}$$

Let $\vec{S}_{\xi} = |J|\nabla\xi$, $\vec{S}_{\eta} = |J|\nabla\eta$, which represent the "area vector" of constant $\xi$ and $\eta$ lines in the physical domain. Obviously $\tilde{F} = \vec{F} \cdot \vec{S}_{\xi}$ and $\tilde{G} = \vec{F} \cdot \vec{S}_{\xi}$. Since the standard elements are linear, the FR/CPR formulation can be directly applied

$$\frac{\partial \tilde{Q}_{i,j}}{\partial t} + \Pi_j(\nabla^{\xi} \cdot \vec{\tilde{F}}(\tilde{Q}_i)) + \frac{1}{|V_i^{\xi}|} \sum_{f \in \partial V_i^{\xi}} \sum_{l} \alpha_{j,f,l} [\tilde{F}^n]_{f,l}^{\xi} S_f^{\xi} = 0, \tag{2.72}$$

where superscript $\xi$ means that the variables or operators are evaluated on the computational domain. Note the index $j$ is for the solution points. For the standard triangle, $|V_i^{\xi}| = 1/2$. For face 1, $\vec{n}_1^{\xi} = (0, -1)$, $S_1^{\xi} = 1$, and

$$\begin{aligned} [\tilde{F}^n]_{1,l}^{\xi} S_1^{\xi} &= -(\tilde{G}_{com} - \tilde{G}(\tilde{Q}_i))_{1,l} \\ &= (F_{com}^n - F^n(Q_i))_{1,l} |\vec{S}_{\eta}|_{1,l} \\ &= [F^n]_{1,l} |\vec{S}_{\eta}|_{1,l}. \end{aligned} \tag{2.73}$$

Similar formulas can be obtained for face 2 and 3. Take into account that

$$\frac{1}{|J|} \nabla^{\xi} \cdot \vec{\tilde{F}} = \nabla \cdot \vec{F}, \tag{2.74}$$

Eq. (2.72) can be further expressed as

$$\frac{\partial Q_{i,j}}{\partial t} + \Pi_j(\nabla \cdot \vec{F}(Q_i)) + \frac{2}{|J|_{i,j}} \sum_{f \in \partial V_i} \sum_{l} \alpha_{j,f,l} [F^n]_{f,l} S_f = 0, \tag{2.75}$$

where $S_{1,l} = |\vec{S}_\eta|_{1,l}$, $S_{2,l} = |\vec{S}_\xi + \vec{S}_\eta|_{2,l}$, and $S_{3,l} = |\vec{S}_\xi|_{3,l}$. The formula for quadrilateral elements can be derived in the same manner. A simpler way to derive the formula is based on the fact that the operations on a standard quadrilateral can be decomposed into the tensor product of one-dimensional operations. For 1D conservation laws, Eq. (2.75) reduces to

$$\frac{\partial Q_{i,j}}{\partial t} + \Pi_j(\nabla \cdot \vec{F}(Q_i)) + \frac{1}{h_i}(\alpha_{L,j}[F^n]_L + \alpha_{R,j}[F^n]_R) = 0, \tag{2.76}$$

where $h_i$ is the length of element $i$, subscripts $L$ and $R$ denotes the value is taken at the two ends of the 1D line, $\alpha_{L,j}$ and $\alpha_{R,j}$ are constant lifting coefficients in 1D. Due to the tensor product, two indices $(j,m)$ are employed to denote the solution point. The CPR formulation for a quadrilateral is then

$$\begin{aligned} &\frac{\partial \tilde{Q}_{i:j,m}}{\partial t} + \Pi_{j,m}(\nabla^\xi \cdot \vec{\tilde{F}}(\tilde{Q}_i)) - [\tilde{F}_{com}(-1,\eta_{j,m}) - \tilde{F}_i(-1,\eta_{j,m})]\frac{\alpha_{L,j}}{2} \\ &+[\tilde{F}_{com}(1,\eta_{j,m}) - \tilde{F}_i(1,\eta_{j,m})]\frac{\alpha_{R,j}}{2} - [\tilde{G}_{com}(\xi_{j,m},-1) - \tilde{G}_i(\xi_{j,m},-1)]\frac{\alpha_{L,m}}{2} \\ &+[\tilde{G}_{com}(\xi_{j,m},1) - \tilde{G}_i(\xi_{j,m},1)]\frac{\alpha_{R,m}}{2} = 0. \end{aligned} \tag{2.77}$$

Note that the correction for quadrilateral cells is done in "one dimension" manner, making the method more efficient per DOF than for triangular cells.

## 2.4 FR/CPR Method for Viscous Flow

For viscous flows, a viscous flux term is added to the Euler equations such that Eq. (2.44) becomes the Navier-Stokes equations,

$$\frac{\partial Q}{\partial t} + \nabla \cdot \vec{F}(Q) = \nabla \cdot \vec{F}^v(Q, \nabla Q), \tag{2.78}$$

where $\vec{F}^v(Q,\nabla Q)$denotes the viscous flux vector. Let

$$\vec{R} = \nabla Q, \tag{2.79}$$

and let $\vec{R}_i$ be an approximate of $\vec{R}$ on $V_i$. The components of $\vec{R}_i$ are required to belong to $\mathbf{P}_k$. Many studies have shown that the obvious choice of $\vec{R}_i = \nabla Q_i$ is not appropriate. Instead, the computation of $\vec{R}_i$ needs to involves data from neighboring cells. With the definition of $\vec{R}$, the CPR formulation on a linear element can be express as

$$\frac{\partial Q_{i,j}}{\partial t} + \Pi_j(\nabla \cdot \vec{F}(Q_i)) - \Pi_j^{\nu}(\nabla \cdot \vec{F}^{\nu}(Q_i, \vec{R}_i)) + \frac{1}{|V_i|} \sum_{f \in \partial V_i} \sum_{l} \alpha_{j,f,l}([F^n]_{f,l} - [F^{\nu,n}]_{f,l}) S_f = 0, \quad (2.80)$$

$$\vec{R}_{i,j} = \nabla Q_{i,j} + \frac{1}{|V_i|} \sum_{f \in \partial V_i} \sum_{l} \alpha_{j,f,l} [Q^{com} - Q_i]_{f,l} \vec{n}_f S_f, \quad (2.81)$$

where $\Pi^{\nu}$ is the projection operator for the divergence of the viscous flux vector to $\mathbf{P}_k$, and

$$[F^{\nu,n}]_f \equiv \vec{F}^{\nu}(Q_f^{com}, \nabla Q_f^{com}) \cdot \vec{n}_f - \vec{F}^{\nu}(Q_i, \vec{R}_i)\big|_f \cdot \vec{n}_f \quad (2.82)$$

with $Q_f^{com}$ and $\nabla Q_f^{com}$ the common solution and its gradient on the interface $f$. The computing of $\Pi_j^{\nu}(\nabla \cdot \vec{F}^{\nu}(Q_i, \vec{R}_i))$ follows the LP approach similar to Eq. (2.62)

$$\Pi_j^{\nu}(\nabla \cdot \vec{F}^{\nu}(Q_i, \vec{R}_i)) \approx \sum_j \vec{F}_{i,j}^{\nu} \cdot \nabla L_j^{SP}. \quad (2.83)$$

Various schemes were proposed to compute the common solution $Q_f^{com}$ and the common gradient $\nabla Q_f^{com}$. In the following subsection, the Bassi-Rebay 2 (BR2) (Bassi & Rebay, 2000), I-continuous (Huynh, 2009), interior penalty (Dolejší, 2004; Hartmann & Houston, 2006), and compact discontinuous Galerkin (CDG) schemes (Peraire & Persson, 2008) are reviewed. For values at the cell interfaces, superscript $-$ and $+$ are retrieved to represent the interior and exterior of the cell respectively. The cell is always considered as the left cell of that face, and the neighbor cell which holds the values with superscript $+$ is always considered as the right cell.

### 2.4.1 Bassi-Rebay 2

The common solution of flux point $l$ in BR2 is simply the average of the solutions at both sides of the face

$$Q_{f,l}^{com} = \frac{Q_{f,l}^{-} + Q_{f,l}^{+}}{2}. \tag{2.84}$$

The common gradient is computed as the average of corrected gradients of the solutions

$$\nabla Q_{f,l}^{com} = \frac{1}{2}(\nabla Q_{f,l}^{-} + \vec{r}_{f,l}^{-} + \nabla Q_{f,l}^{+} + \vec{r}_{f,l}^{+}), \tag{2.85}$$

where $\nabla Q_{f,l}^{-}$ and $\nabla Q_{f,l}^{+}$ are the gradients of the solutions at the left and right cells without correction, $\vec{r}_{f,l}^{-}$ and $\vec{r}_{f,l}^{+}$ are corresponding corrections:

$$\begin{aligned} \vec{r}_{f,l}^{-} &= \frac{1}{|V^{-}|} \sum_{m=1}^{N_{FP}} \beta_{l,m} [Q^{com} - Q^{-}]_{f,m} \vec{n}_f S_f, \\ \vec{r}_{f,l}^{+} &= \frac{1}{|V^{+}|} \sum_{m=1}^{N_{FP}} \beta_{l,m} [Q^{com} - Q^{+}]_{f,m} (-\vec{n}_f) S_f, \end{aligned} \tag{2.86}$$

where $N_{FP}$ is the number of flux points on face $f$, $\beta_{l,m}$ is the coefficient of correction due to face $f$. Note that $l$ and $m$ vary on face $f$, one choice is $\beta_{l,m} = \alpha_{j,f,m}$, where solution point $j$ corresponds to the flux point $l$ on the face.

### 2.4.2 I-Continuous

The basic idea of the I-continuous scheme is to treat $Q^{com}$ as an unknown instead of prescribing it. This unknown is solved by the condition that the corrected gradients are continuous in the normal

direction across the interface. The corrected gradient on both sides of face $f$ can be expressed as

$$\begin{aligned}\nabla Q_{f,l}^{com-} &= \nabla Q_{f,l}^{-} + \frac{1}{|V^-|}\sum_{m=1}^{N_{FP}} \beta_{l,m}[Q^{com} - Q^-]_{f,m}\vec{n}_f S_f, \\ \nabla Q_{f,l}^{com+} &= \nabla Q_{f,l}^{+} + \frac{1}{|V^+|}\sum_{m=1}^{N_{FP}} \beta_{l,m}[Q^{com} - Q^+]_{f,m}(-\vec{n}_f) S_f.\end{aligned} \tag{2.87}$$

The requirement of continuous gradient in normal direction gives

$$\nabla Q_{f,l}^{com-} \cdot \vec{n}_f = \nabla Q_{f,l}^{com+} \cdot \vec{n}_f. \tag{2.88}$$

Substituting Eq. (2.87) into Eq. (2.88), we obtain

$$\sum_{m=1}^{N_{FP}} \left(\frac{\beta_{l,m}}{|V^-|} + \frac{\beta_{l,m}}{|V^+|}\right) Q_{f,m}^{com} S_f = (\nabla Q_{f,l}^{+} - \nabla Q_{f,l}^{-}) \cdot \vec{n}_f + \sum_{m=1}^{N_{FP}} \beta_{l,m} \left(\frac{Q_{f,m}^{-}}{|V^-|} + \frac{Q_{l,m}^{+}}{|V^+|}\right) S_f. \tag{2.89}$$

Above equation is a linear system with unknown $Q_{f,m}^{com}, m = 1, \cdots, N_{FP}$. It can be solved be any solver for linear systems. Note the matrix at the left hand side of above linear system is independent of the solutions, it only needs to be inverted once at the initializing stage. Therefore, the I-continuous scheme can be almost as efficient as the BR2 scheme. Substituting the common solution back to Eq. (2.87), the gradient of the common solution is obtained.

### 2.4.3 Interior Penalty

The interior penalty scheme can be seen as a simplified version of BR2. In Eq. (2.86) of the BR2, the correction terms at the flux point $l$ is the linear combination of the solution differences at all flux points on face $f$. In interior penalty scheme, the corrections or the penalty is only restricted to

the solution difference at that flux point, i.e.

$$\begin{aligned}\vec{r}_{f,l}^{-} &= \frac{\beta_{f,l} S_f}{|V^-|}[Q^{com} - Q^-]_{f,l}\vec{n}_f, \\ \vec{r}_{f,l}^{+} &= -\frac{\beta_{f,l} S_f}{|V^+|}[Q^{com} - Q^+]_{f,l}\vec{n}_f,\end{aligned} \tag{2.90}$$

where $\beta_{f,l}$ is a constant for any $l$.

### 2.4.4 Compact Discontinuous Galerkin

Simmilar to the local DG approach (Cockburn & Shu, 1998), the CDG scheme uses the solution on one side as the common solution, and correctied gradient of the other side as the common gradient. CDG is compact for arbitrary unstructured meshes, while the local DG may not be.

If the right side is used for the common solution and the left side is used for the common gradient, the CDG gives

$$\begin{aligned}Q_{f,l}^{com} &= Q_{f,l}^{+}, \\ \nabla Q_{f,l}^{com} &= \nabla Q_{f,l}^{-} + \vec{r}_{f,l}^{-},\end{aligned} \tag{2.91}$$

where

$$\vec{r}_{f,l}^{-} = \frac{1}{|V^-|}\sum_{m=1}^{N_{FP}} \beta_{l,m}[Q^{com} - Q^-]_{f,m}\vec{n}_f S_f. \tag{2.92}$$

Alternatively, common solution and its gradient can be calculated by the values on the opposite side:

$$\begin{aligned}Q_{f,l}^{com} &= Q_{f,l}^{-}, \\ \nabla Q_{f,l}^{com} &= \nabla Q_{f,l}^{+} + \vec{r}_{f,l}^{+},\end{aligned} \tag{2.93}$$

where

$$\vec{r}_{f,l}^{+} = \frac{1}{|V^+|}\sum_{m=1}^{N_{FP}} \beta_{l,m}[Q^{com} - Q^+]_{f,m}\vec{n}_f S_f. \tag{2.94}$$

## 2.5 FR/CPR Method for Dynamic Moving Grid

Since most of the computations of the FR/CPR method is in the computational domain, it is naturally applying the method for moving grids in computational domain. The transformation from the physical domain to the computational domain is now depending on both space and time, i.e. $(x,y,t) \rightarrow (\xi,\eta,\tau)$, where $\tau = t$. Instead of Eq. (2.68), the Jacobian matrix takes the following form

$$J = \frac{\partial(x,y,t)}{\partial(\xi,\eta,\tau)} = \begin{bmatrix} x_\xi & x_\eta & x_\tau \\ y_\xi & y_\eta & y_\tau \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.95}$$

and the inverse of the Jacobian matrix

$$J^{-1} = \frac{\partial(\xi,\eta,\tau)}{\partial(x,y,t)} = \begin{bmatrix} \xi_x & \xi_y & \xi_t \\ \eta_x & \eta_y & \eta_t \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.96}$$

The variable at the subscript means taking derivative with respect to that variable. The conservation laws in the computational domain can be written as

$$\frac{\partial \tilde{Q}}{\partial \tau} + \frac{\partial \tilde{F}}{\partial \xi} + \frac{\partial \tilde{G}}{\partial \eta} = 0, \tag{2.97}$$

where

$$\begin{cases} \tilde{Q} = |J|Q, \\ \tilde{F} = |J|(\xi_x F + \xi_y G + Q\xi_t), \\ \tilde{G} = |J|(\eta_x F + \eta_y G + Q\eta_t). \end{cases} \tag{2.98}$$

Note that the information of grid velocity $\vec{v}_g = (x_t, y_t)$ is contained in $\xi_t$ and $\eta_t$:

$$\begin{cases} \xi_t = -\vec{v}_g \cdot \nabla\xi, \\ \eta_t = -\vec{v}_g \cdot \nabla\eta. \end{cases} \tag{2.99}$$

The summation of the terms that involves $\xi_t$ and $\eta_t$ in Eq. (2.97) can be written as

$$\frac{\partial}{\partial \xi}(Q|J|\xi_t) + \frac{\partial}{\partial \eta}(Q|J|\eta_t) = \nabla^{\xi} \cdot (Q|J|\frac{\partial \vec{\xi}}{\partial t}) \tag{2.100}$$

Transforming above expression back to the physical domain, we obtain a term $\nabla \cdot (-Q\vec{v}_g)$. The $-Q\vec{v}_g$ carries a physical meaning, that is, a flux vector caused by the motion of the cell.

When the cell is moving in a freestream, Eq. (2.97) is required to be strictly satisfied such that the solution is not changing with time. This is known as the geometric conservation law (GCL). Since the GCL applys to any constant $Q$, $F$, and $G$, according to Eq. (2.97), the following equations hold

$$\begin{cases} \dfrac{\partial}{\partial \xi}(|J|\xi_x) + \dfrac{\partial}{\partial \eta}(|J|\eta_x) = 0, \\ \dfrac{\partial}{\partial \xi}(|J|\xi_y) + \dfrac{\partial}{\partial \eta}(|J|\eta_y) = 0, \\ \dfrac{\partial |J|}{\partial t} + \dfrac{\partial}{\partial \xi}(|J|\xi_t) + \dfrac{\partial}{\partial \eta}(|J|\eta_t) = 0. \end{cases} \tag{2.101}$$

Note the first two equations are only involved with the spatial transformation. They are automatically satisfied since the spatial transformation is exact. It is obvious by chain rule that

$$\frac{\partial \tilde{Q}}{\partial t} = \frac{\partial (|J|Q)}{\partial t} = |J|\frac{\partial Q}{\partial t} + Q\frac{\partial |J|}{\partial t}. \tag{2.102}$$

Substitute the last equation of Eq. (2.101) into above equation, we obtain

$$\frac{\partial \tilde{Q}}{\partial t} = |J|\frac{\partial Q}{\partial t} - Q\left[\frac{\partial}{\partial \xi}(|J|\xi_t) + \frac{\partial}{\partial \eta}(|J|\eta_t)\right] \tag{2.103}$$

The Eq. (2.97) is thus changed to the following form

$$\frac{\partial Q}{\partial t} + \frac{1}{|J|}\left\{\left(\frac{\partial \tilde{F}}{\partial \xi} + \frac{\partial \tilde{G}}{\partial \eta}\right) + \text{Source}\right\} = 0, \tag{2.104}$$

where

$$\text{Source} = -Q\left[\frac{\partial}{\partial\xi}(|J|\xi_t)+\frac{\partial}{\partial\eta}(|J|\eta_t)\right] \tag{2.105}$$

Note that $\frac{\partial\tilde{F}}{\partial\xi}+\frac{\partial\tilde{G}}{\partial\eta}$ contains a term $\frac{\partial}{\partial\xi}(Q|J|\xi_t)+\frac{\partial}{\partial\eta}(Q|J|\eta_t)$ (left hand side of Eq. (2.100)). This term will be canceled by the "source" term when $Q$ is a constant. Therefore the GCL is strictly satisfied. A benefit of adding the source term is that the calculation of $\frac{\partial|J|}{\partial t}$ is avoid, which is computational expensive.

The source term can then be transformed back into the physical domain

$$-Q\left[\frac{\partial}{\partial\xi}(|J|\xi_t)+\frac{\partial}{\partial\eta}(|J|\eta_t)\right] = -Q\nabla^{\xi}\cdot\left(|J|\frac{\partial\vec{\xi}}{\partial t}\right) = Q\nabla\cdot\vec{v}_g. \tag{2.106}$$

Combining the source term with the gradient of the flux term causing by the moving grid, i.e. $\nabla(-Q\vec{v}_g)$ in physical domain, a new term is obtained

$$\nabla(-Q\vec{v}_g)+Q\nabla\vec{v}_g = -\nabla Q\cdot\vec{v}_g. \tag{2.107}$$

Adding this term to Eq. (2.80) and modifying the interface flux jump, the formula of the FR/CPR method for moving grids can be expressed in physical domain as

$$\begin{aligned}&\frac{\partial Q_{i,j}}{\partial t}+\Pi_j(\nabla\cdot\vec{F}(Q_i)-\nabla Q_i\cdot\vec{v}_{g:i,j})-\Pi_j^{\nu}(\nabla\cdot\vec{F}^{\nu}(Q_i,\vec{R}_i))\\&+\frac{1}{|V_i|}\sum_{f\in\partial V_i}\sum_{l}\alpha_{j,f,l}([F^n]_{f,l}-[Q^{com}-Q_i]_{f,l}\vec{v}_{g:f,l}\cdot\vec{n}_f-[F^{\nu,n}]_{f,l})S_f=0,\end{aligned} \tag{2.108}$$

where $\vec{v}_{g:i,j}$ is the grid velocity at the solution point $j$ of cell $i$, and $\vec{v}_{g:f,l}$ is the grid velocity at the flux point $l$ on face $f$. The grid velocity is generally changing with time. Inappropriate calculation of the grid velocity in a time scheme could introduce dramatic errors into the simulation. We will address this in Chapter 4.

# Chapter 3

# Overset Assembly

This chapter provides details of the overset FR/CPR method which is developed as part of this dissertation. Different approaches for high-order data communications between overset grids are presented. A hole cutting method based on auxiliary approximate geometries is introduced. An efficient approach for building the receptor-donor connectivity is developed. The sliding mesh method is also presented as a special case of moving overset meshes.

It is helpful to define some terminologies before we start the discussion.

- *Hole cells*: cells that have been removed from the consideration of the numerical scheme and therefore do not take part in any interpolation procedure.
- *Hole points/nodes*: points or mesh nodes that do not take part in any interpolation procedure.
- *Receptor points*: points at which the solutions are interpolated from an overlapping cell.
- *Receptor cells*: cells that host the receptor points.
- *Donor cells*: cells that provide interpolated solutions for receptor points.

## 3.1 High-Order Data Communication Approaches

Since the high-order DG methods were developed for overset unstructured grids, two main approaches are incorporated to handle data communication between the grids. One is a face-based interpolation approach developed in (Galbraith et al., 2015), and the other is an element-based interpolation approach developed in (Nastase et al., 2011; Brazell et al., 2016). Although DG

methods often use "modes" of polynomials as the basis functions without explicitly defining physical points for the polynomials in a cell, above overset DG methods define a set of quadrature points at each receptor cell. These points are receptor points. In the face-based approach, the receptor points are defined on certain faces of the receptor cell where interpolated solutions are needed. The interpolated nodal solution values are converted to solution polynomials on those faces, and then the polynomials are used as boundary conditions weakly applied through a Riemann solver. Therefore, this approach is also known as artificial boundary approach. In the element-based approach, the receptor points are defined at the interior of the receptor cell. The interpolated nodal solution values are directly converted to the modes of the receptor cell. In the FR/CPR method, since the solution points and the flux points are defined to store the solutions, it is pretty straightforward to interpolate solutions at these points. Figure 3.1 shows an example of the face-based interpolation

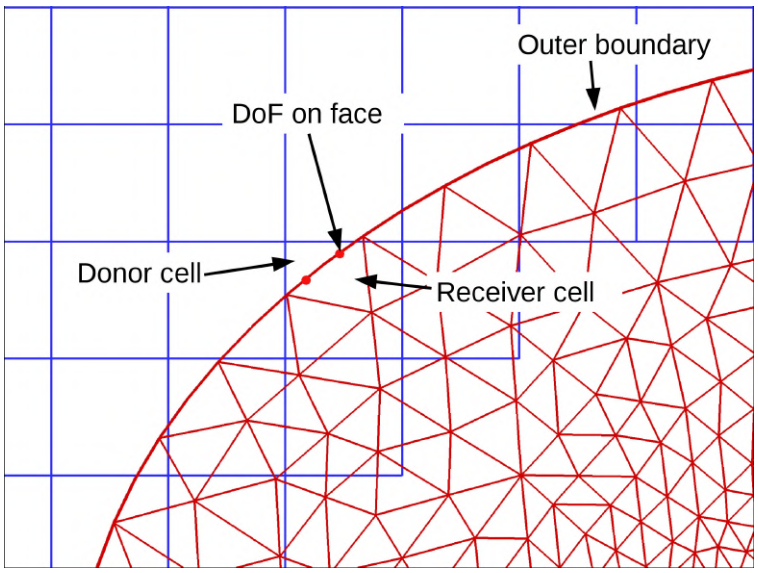



Figure 3.1: Face-based interpolation approach. Flux points (red dots) on the outer boundary receive interpolated solutions from the donor cell in the Cartesian mesh.

approach. The flux points (red points) on the boundary face of the receptor cell are used as receptor points. These receptor points receive interpolated point values from their donor cells. For a receptor point, let $(\xi, \eta, \zeta)$ be its coordinates in the mapped standard element of the donor cell.

The solution $Q$ is interpolated by

$$Q^R(\xi,\eta,\zeta) = \sum_{j=1}^{N_{SP}} \phi_j^D(\xi,\eta,\zeta) Q_j^D, \tag{3.1}$$

where $j$ is the index of the solution points of the donor cell, $N_{SP}$ is the number of solution points, $\phi_j^D(\xi,\eta,\zeta)$ are the basis functions of the donor cell, and $Q_j^D$ are the solution values at the donor cell's solution points. As a convention, the superscript "$D$" denotes for the donor and "$R$" denotes for the receptor. The interpolated solutions are then used as right values at the boundary face of the receptor to compute the common flux

$$F_{com}^n = F_{l,Riemann}(Q_l^{R-}, Q_l^{R+}), \tag{3.2}$$

where $F_{Riemann}$ is a Riemann solver, $l$ is the index of the flux point on the face, $Q_l^{R-}$ is the left value which is computed from the internal DoFs of the receptor cell, $Q_l^{R+}$ is the right value computed by Eq. (3.1). The common fluxes are then used by the FR/CPR method to update the solutions of the receptor cells.

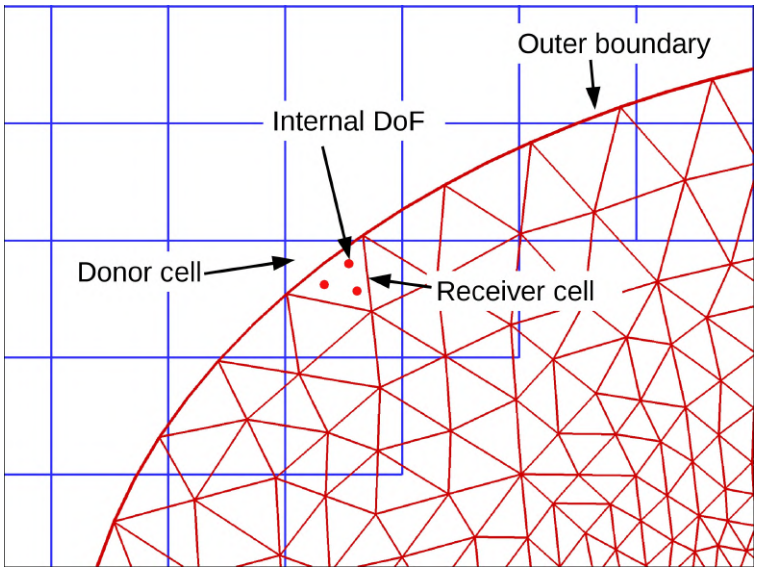



Figure 3.2: Element-based interpolation approach. Solution points (red dots) inside the receptor cell receive data from the donor cell in the Cartesian mesh.

The element-based interpolation approach is shown in Figure 3.2. The solution points (red

points) at the interior of the receptor cell are used as receptor points. The interpolated solutions are computed in the same way as the face-based approach, i.e. Eq. (3.1). Note that the solutions of the receptor cell are directly interpolated from the donor cells without going through the FR/CPR method. However the solutions at the donor cells are updated by the FR/CPR method. The donor cells then provide interpolations for the receptor cells to update. We will compare the stability and accuracy of the two data communication approaches in Chapter 5.

## 3.2 Hole Cutting

An Overset system can have various configurations. The most commonly seen overset system in CFD simulations perhaps is the one with one mesh entirely embedded into another mesh. The embedded mesh often has a solid body inside, and is therefore called the near-body mesh or simply body mesh. The containing mesh is correspondingly called the off-body mesh. Since the off-body mesh is typically much larger in physical scale than the near-body mesh, it is also called the background mesh. Another overset configuration is that multiple meshes overlap each other without one mesh entirely embedding into any other mesh. Overset meshes with this configuration are often referred as the chimera meshes. For complex geometries, the configuration could be the combination of previous two overset configurations. For example multiple partially overlapped near-body meshes are embedded in one background mesh. Depending on the overset system, the hole cutting methods could be various. Before introducing the hole cutting method for the work of this dissertation, different types of hole cutting methods are reviewed below.

### 3.2.1 Hole Cutting Methods Review

The hole cutting methods can be roughly divided into three types (Noack, 2007). The first type, namely the search-based methods, is characterized by the use of fast search methods such as a KD-tree or an Alternating Digital Tree (ADT) (Bonet & Peraire, 1991). A search-based method typically attempts to find a donor cell from another mesh for a node of the current mesh. If no donor

cell is found, that node is considered as a potential hole node. The potential hole node is further checked if it lies inside some approximate geometry representations of solid walls, or if its location relative to the wall exceeds some limitations. An approximate geometry representation can be a Cartesian approximate map. Alternatively, methods such as the implicit hole cutting method (Lee & Baeder, 2003) try to find median lines between wall boundaries of different meshes. Points that are inside the approximate geometry or locate beyond the median lines are determined to be hole points. This type of methods combines hole cutting and receptor-donor connectivity into one single process.

The second type of hole cutting methods, query-base methods, also use an approximate geometry representation of the meshes. The approximate geometry must be easily queried to determine if a node or cell of one mesh lies inside a wall boundary of another mesh. Examples of the approximate geometry include Cartesian grid approximations, quad/octree grid representations, and other binary tree representations. As long as the approximate geometry is easily constructed, this type of hole cutting method can be very fast and efficient. However cares need to be taken when the geometry is complex or has small features such as a small gap between two bodies or between a wall boundary and its enclosing outer boundary. Additional information and data structures can be used to relieve these difficulties (Sitaraman et al., 2010; Roget & Sitaraman, 2014).

The last type of hole cutting methods are direct cut methods (Noack, 2007). The boundary surfaces of a grid are directly used to 'cut' through other grids and remove all cells inside the boundary. Fast binary search trees are still necessary to locate the cells to be cut. The key of this type of methods is finding the face-cell intersection efficiently and correctly. It is challenging for the direct cut methods when high-order curved elements present in the grids (Galbraith, 2013). Since high-order curved elements is preferred for high-order finite element methods, a recent work tried to setup an efficient parallel direct cut method for meshes with curved elements (Crabill et al., 2018). By using GPUs the new method shows good parallel efficiency.

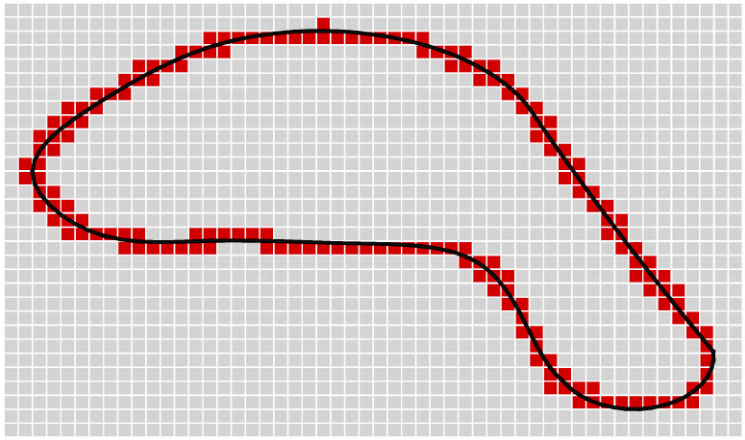
### 3.2.2 Hole Cutting for the Overset FR/CPR Method

The hole cutting approach we are using is a query-based method that follows the approach developed in (Roget & Sitaraman, 2014). This approach is designed for efficiently implement in parallel environment. The approximate geometry is an auxiliary Cartesian grid (ACG). For each near-body mesh, a unique ACG is built and associated with the body mesh. The auxiliary grid is generated from the bounding box of the near body mesh. Figure 3.3 shows an example of the outer boundary of a near-body mesh and its associated auxiliary grid in 2D. The black curve represents the outer boundary and the white Cartesian grid is the auxiliary grid. To build the ACG, firstly the domain size and the dimensions need to be determined. The bounding box of the near-body mesh is defined from the maximum and minimum coordinates of the mesh nodes, $(x_{min}, y_{min}, z_{min}, x_{max}, y_{max}, z_{max})$. The average cell size $(d\hat{x}, d\hat{y}, d\hat{z})$ at the outer boundary of the near-body mesh can also be obtained when the bounding box is defined. In parallel environment, each process only has a piece of the near-body mesh stored locally. A process searches through its local piece of the outer boundary faces of the near-body mesh and finds a local bounding box for these faces. The local average cell size can also be determined upon the cells attached to those faces. After each process finds the local bounding box and the local average cell size, these information is broadcasted among all processes such that the global bounding box and the average cell size is determined. Note that this communication does not require to collect all faces of the outer boundary. The dimensions of the auxiliary Cartesian grid are then determined according to

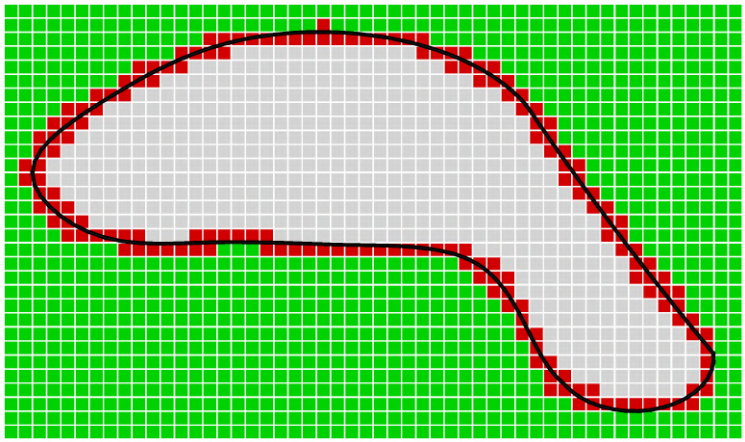
$$N_x = \left\lceil \frac{x_{max} - x_{min}}{d\hat{x}} \right\rceil, \quad N_y = \left\lceil \frac{y_{max} - y_{min}}{d\hat{y}} \right\rceil, \quad N_z = \left\lceil \frac{z_{max} - z_{min}}{d\hat{z}} \right\rceil, \tag{3.3}$$

where $\lceil \cdot \rceil$ is an operator that returns an integer of the upper bound of a real number. The cell size of the auxiliary Cartesian grid $(dx, dy, dz)$ can be computed from

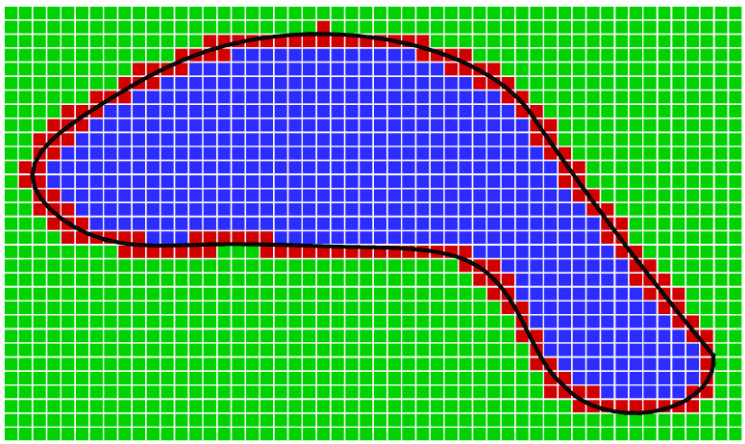
$$dx = \frac{x_{max} - x_{min}}{N_x}, \quad dy = \frac{y_{max} - y_{min}}{N_y}, \quad dz = \frac{z_{max} - z_{min}}{N_z}. \tag{3.4}$$



(a) find the elements (red) of the auxiliary grid that are intersected by the outer boundary of the body mesh.



(b) mark the elements (green) that are outside the outer boundary.



(c) remaining elements (blue) are inside the outer boundary of the body mesh.

Figure 3.3: An example of building a holemap for an auxiliary grid in 2D. Black curve: the outer boundary of the near body mesh; White grid: the auxiliary grid; Grey elements: unmarked elements

To distinguish the cells of the auxiliary Cartesian grid from the cells of the overset meshes, we will call the cells of the auxiliary grid as blocks instead of cells from now on. To mark all the blocks inside the near-body mesh and all the blocks outside the near-body mesh with different colors, it is necessary to add one extra layer of blocks on the boundary of the auxiliary grid. We will explain the reason later in this section. Therefore $(N_x, N_y, N_z)$ needs to be increased by 2 in each direction. It is worth noting that the parameters of the entire auxiliary Cartesian grid, i.e. $(x_{min}, y_{min}, z_{min}, x_{max}, y_{max}, z_{max})$ and $(N_x, N_y, N_z)$, are compact enough to be distributed to all processors in a parallel implementation. This property allows the auxiliary Cartesian grid to be set up very efficiently. For any given point $P = (x, y, z)$ located inside the auxiliary Cartesian grid, the indices of the containing block are immediately calculated as

$$i = \left\lfloor \frac{x - x_{min}}{dx} \right\rfloor, \quad j = \left\lfloor \frac{y - y_{min}}{dy} \right\rfloor, \quad k = \left\lfloor \frac{z - z_{min}}{dz} \right\rfloor, \tag{3.5}$$

where $\lfloor \cdot \rfloor$ is an operator that returns an integer of the lower bound of a real number. For a given cell and/or face of the overset meshes, the coordinates of its nodes are also known. By using Eq. (3.5) one can rapidly find out the blocks that intersect the cell and/or face. Here we use an example to illustrate this. Suppose a face on the outer boundary of the near-body mesh has $n$ nodes. Their coordinates are $(x_l, y_l, z_l)$, $l = 1 \cdots n$. For node $l$, the indices of the containing block $(i_l, j_l, k_l)$ are found by Eq. (3.5). Therefore, the minimum indices $(i_{min}, j_{min}, k_{min})$ and the maximum indices $(i_{max}, j_{max}, k_{max})$ are known when $l$ traverse through all the nodes. Any block whose indices simultaneously fall inside the ranges $[i_{min}, i_{max}]$, $[j_{min}, j_{max}]$, and $[k_{min}, k_{max}]$ is considered as intersecting the face on the outer boundary. Using the faces on the outer boundary of the near body mesh, a subset of blocks which intersect that boundary is found. These blocks are tagged as "intersected blocks", as shown in Figure 3.3(a), colored red blocks. Once the intersected blocks are marked out, the blocks inside the near body boundary need to be distinguished from those outside. This can be achieved using the flood-fill algorithms. Since an extra layer of blocks has been added to the auxiliary Cartesian grid, the outmost layer of the blocks is definitely outside the near-body

mesh. The blocks of this layer serve as the seeds so that the flood-fill algorithms can proceed. The flood-fill algorithms keep on tagging unmarked neighboring blocks as "exterior blocks" until it hits the intersected blocks. Figure 3.3(b) shows the result of the flood-fill algorithms. All the blocks outside the outer boundary of the near-body mesh are tagged as exterior blocks and colored as green. The rest uncolored blocks are then tagged as "interior blocks" (blue blocks in Figure 3.3(c)). Til now, all the blocks of the ACG are marked with different tags/colors. This process is called hole profiling.

The next step of the hole cutting process is to find the cells of the background mesh that are located inside the near-body mesh. Since all the blocks have been tagged, these information will be used to determine if or not a given cell of the background mesh is located inside the near-body mesh. From the nodes' coordinates of the cell, the ranges of block indices, i.e. $[i_{min}, i_{max}]$, $[j_{min}, j_{max}]$, and $[k_{min}, k_{max}]$, can be computed by Eq. (3.5). We name the blocks within these ranges as attached blocks of the cell. The background cells can be divided into four categories by its attached blocks:

1. Definite exterior: the attached blocks are all exterior blocks, or no attached block exists because of the background cell locating outside of the ACG.

2. Potential intersect: the attached blocks include intersected blocks but no interior blocks.

3. Potential interior: the attached blocks include both interior blocks and other types of blocks.

4. Definite interior: the attached blocks are all interior blocks.

It is straightforward to consider only the definite interior cells as inside the near-body mesh. These cells are marked as inactive hole cells, and the rest background cells are marked as active cells, of which the flow solver only updates the solutions. However, this approach can be failed. In a near-body mesh, the outer boundary could be close to the enclosed wall boundary. It is possible that some background cells which belongs to potential interior cells are not cut out, such that they intersect the solid body. These cells will find no donor cells. To reduce the chance of this

problem appearing, one additional step which checks the potential interior cells is performed. For each potential interior cell of the background mesh, an oriented bounding box (OBB) is built. A collision test between the OBB and the attached blocks is performed to find out those blocks that are intersected with the OBB. This process will generally reduce the number of attached blocks by ruling out the blocks that do not intersect the OBB. If all the remaining attached blocks are interior blocks, the background cell is marked as an inactive hole cell. This additional step will generally cut out more cells than the definite interior cells, and therefore reduces the chance of the active cells intersecting the solid body. If these cells which intersect the solid body still remain after the hole cutting, either the ACG or the background mesh needs to be refined, which is out the scope of this dissertation. Figure 3.4 shows an example of an OBB of a 6-nodes curved triangle. The $O(x,y)$ is the Cartesian coordinate system and the $O'(x',y')$ is the local coordinate system. To build the OBB of a cell, the cell center $(x_c, y_c, z_c)$ needs to be found first:

$$\begin{cases} x_c = \sum_i^n x_i/n, \\ y_c = \sum_i^n y_i/n, \\ z_c = \sum_i^n z_i/n \end{cases} \tag{3.6}$$

where $i$ is the index of a node and $n$ is the number of nodes in that cell. The axes of the OBB are chosen to be the eigenvectors of the covariance matrix of the nodes' coordinates. The covariance matrix $\mathbf{K}$ is computed as

$$\mathbf{K} = \begin{pmatrix} E[(x-x_c)^2] & E[(x-x_c)(y-y_c)] & E[(x-x_c)(z-z_c)] \\ E[(y-y_c)(x-x_c)] & E[(y-y_c)^2] & E[(y-y_c)(z-z_c)] \\ E[(z-z_c)(x-x_c)] & E[(z-z_c)(y-y_c)] & E[(z-z_c)^2] \end{pmatrix}, \tag{3.7}$$

where $E[\cdot]$ is an operator to compute the average value. The eigenvectors with unit length are denoted as $\hat{x}'$ and $\hat{y}'$ to represent the axes of the OBB. The location vector of each node is then projected onto these axes such that the minimum and maximum coordinates of the projected points,

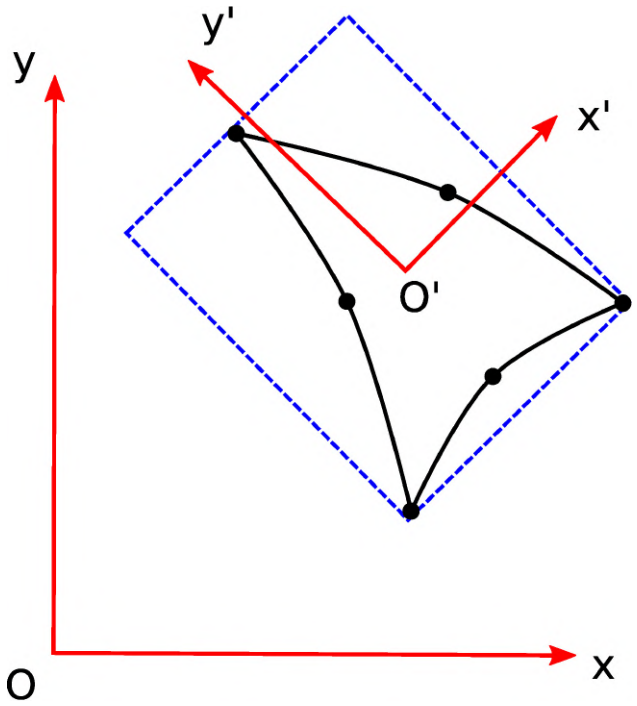



Figure 3.4: An oriented bounding box (blue box with dashed edges) of a 6-nodes curved triangle.

$\{x_{min}^{proj}, x_{max}^{proj}, y_{min}^{proj}, y_{max}^{proj}\}$ in the Cartesian coordinate system are obtained. The center of the OBB is computed as

$$x_{O'} = \frac{x_{min}^{proj} + x_{max}^{proj}}{2}, \quad \text{and} \quad y_{O'} = \frac{y_{min}^{proj} + y_{max}^{proj}}{2}. \tag{3.8}$$

The widths of the OBB are

$$d_{x'} = \frac{x_{max}^{proj} - x_{min}^{proj}}{2}, \quad \text{and} \quad d_{y'} = \frac{y_{max}^{proj} - y_{min}^{proj}}{2}. \tag{3.9}$$

Note that the center of the OBB generally differs from the center of the cell. To perform the collision test between the OBB of the background cell and its attached blocks, an efficient Separating Axis Theorem (Gottschalk et al., 1996) is employed. The Separating Axis Theorem was originally developed to detect the intersection of convex polygons. It states that two convex polygons do not intersect if and only if there exists a line such that the projections of the two polygons onto the line do not intersect. The line is knows as a separating axis. The principle of the Separating Axis Theorem is illustrated in Figure 3.5. Consider two rectangles A and B in the figure, they do not intersect

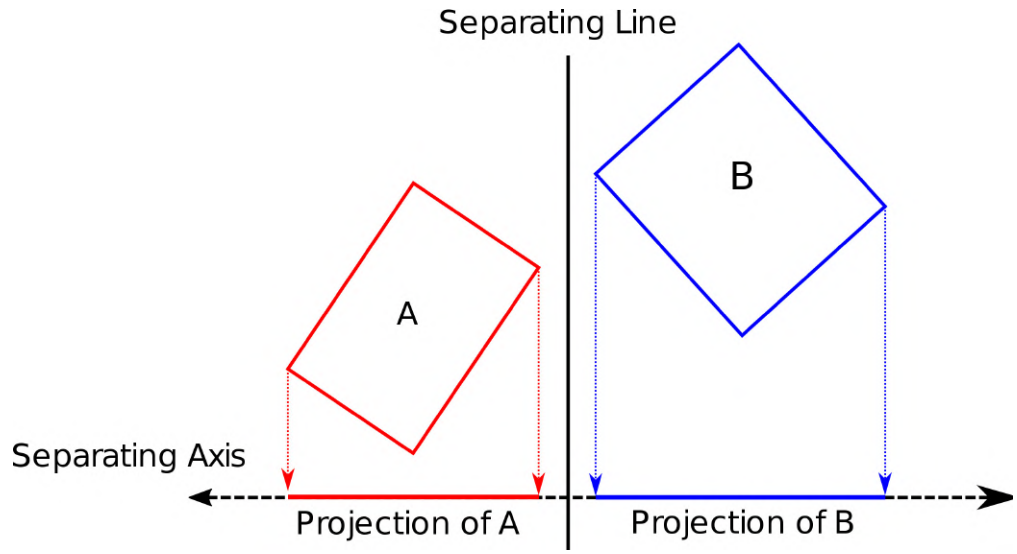



Figure 3.5: Illustration of the Separating Axis Theorem for the collision test of rectangles in 2D space.

because a separating axis is found to the bottom of the rectangles such that the two projections do not overlap. The Separating Axis Theorem is also equivalently defined as two polygons do not intersect if and only if there exists a line that completely divides a polygon on one side of the line and the other polygon on the other side, as shown in Figure 3.5 (black solid line). In 3D space, the separating line is no longer used. Instead, separating planes are defined to divide the bounding volumes. When test the collision between the background cell and the attached blocks, the OBB of a block is simply the block itself with its local axes aligning with the Cartesian axes. After all the background cells are marked, a "hole boundary" in the background mesh is then formed by the interfaces between the inactive cells and active cells. We choose to use the outer boundary rather than the wall boundary to perform hole-cutting for efficiency reasons. In (Roget & Sitaraman, 2014), hole-cutting is performed using wall boundaries.

## 3.3 Receptor-Donor Connectivity

After the hole boundary in the background mesh is cut out, we need to determine the donor cells for the receptor points. This step is also known as donor searching. As mentioned before, the chosen of the receptor points depends on the data communication approach. The receptor points

are the solution points of the cells that attached to the outer boundary of the near-body mesh or the hole boundary of the background mesh, if the element-based approach is chosen. If the face-based approach is chosen, then the receptor points are the flux points of the faces on the outer boundary or the hole boundary. Since the auxiliary Cartesian grid is overlapped with both the outer boundary and the hole boundary, the donor searching can also benefit from the ACG. In previous section each background cell which intersects the ACG is associated with its attached blocks. This builds a map from the cell to the blocks. That is, for a given background cell, a list of index of the attached blocks is generated, i.e. $e_l^{BG} \mapsto \{(i_{l1}, j_{l1}, k_{l1}),\ (i_{l2}, j_{l2}, k_{l2}), \cdots\}$, where $l$ is the index of the cell, $e_l^{BG}$ is the background cell, and $(i_{lm}, j_{lm}, k_{lm})$ are the indices of the attached blocks. A map between the cells of the near-body mesh and the blocks of the ACG can also be built in the same way, i.e. $e_l^{NB} \mapsto \{(i_{l1}, j_{l1}, k_{l1}),\ (i_{l2}, j_{l2}, k_{l2}), \cdots\}$, where $e_l^{NB}$ is a cell of the near-body mesh. The inverse map of these maps can then be setup as $(i, j, k) \mapsto \{e_1^{NB}, e_2^{NB}, \cdots, e_1^{BG}, e_2^{BG}, \cdots\}$. In the inverse map, the cells in the list can belong to either the near-body mesh or the background mesh. For a given receptor point, its physical coordinates $\vec{x} = (x, y, z)$ are known. Through Eq. (3.5) one can immediately locate the index of the containing block. It is possible that the receptor point happens to locate at the interface of two neighboring blocks. In this case, both the blocks are considered to be the containing block. Looking up the inverse map, a list of candidate donor cells can be determined. If the receptor point belongs to the near-body mesh, only the cells in the list that belong to the background mesh are tagged as candidate donors, and vice versa. The inverse map is named as the Exact Inverse Map in (Roget & Sitaraman, 2014). By using the inverse map, the number of candidate donor cells is greatly reduced than searching through every cell in the meshes.

The true donor cell for the given receptor point needs then to be determined from the set of candidate donor cells. To do this, known the physical coordinates of the receptor point $(x, y, z)$, we compute its reference coordinates $\vec{\xi} = (\xi, \eta, \zeta)$ with respect to each candidate donor cell. The

reference coordinates are solved from the following nonlinear equations by Newton's method:

$$\begin{cases} x(\xi,\eta,\zeta) = \sum_j \psi_j(\xi,\eta,\zeta) x_j \\ y(\xi,\eta,\zeta) = \sum_j \psi_j(\xi,\eta,\zeta) y_j \\ z(\xi,\eta,\zeta) = \sum_j \psi_j(\xi,\eta,\zeta) z_j \end{cases}, \tag{3.10}$$

where $j$ is the index the nodes of the candidate donor cell, $(x_j, y_j, z_j)$ are the physical coordinates of node $j$, and $\psi_j(\xi,\eta,\zeta)$ is the mapping basis function defined on node $j$. In our implementation, the basis functions are chosen to be the Lagrange polynomials. If the reference coordinates satisfy the point-containment criteria of a cell, the receptor point is considered to be inside that cell, and the cell is chosen to be the true donor cell. Table 3.1 gives the point-containment criteria for several common 2D and 3D cell types.

Table 3.1: The point-containment criteria for standard cells in 2D/3D space.

| Cell Type | Point-Containment Criteria |
|---|---|
| Triangle | $\xi,\eta \in [0,1]$ , $\xi+\eta \leq 1$ |
| Quadrilateral | $\lvert\xi\rvert \leq 1$, $\lvert\eta\rvert \leq 1$ |
| Hexahedron | $\lvert\xi\rvert \leq 1$, $\lvert\eta\rvert \leq 1$, $\lvert\zeta\rvert \leq 1$ |
| Prism | $\xi,\eta \in [0,1]$ , $\xi+\eta \leq 1$, $\lvert\zeta\rvert \leq 1$ |
| Tetrahedron | $\xi,\eta,\zeta \in [0,1]$, $(\xi+\eta)$, $(\xi+\zeta)$, $(\eta+\zeta) \leq 1$ |

Sometimes the number of candidate donor cells can be big such that performing the Newton's method to solve the Eqs. (3.10) for each of the cells is expensive. This situation often happens when the resolution of the ACG is very coarse comparing to the some region of the overset meshes. One block in ACG can be attached to tens of cells. To further reduce the computational cost, we test if the flux point is contained by the OBB of the candidate cells before the Newton's method is applied. Figure 3.6 shows the possible positions of a receptor point relative to its candidate donor cells. If the receptor point is outside the OBB of the candidate donor cell (point a), the cell is

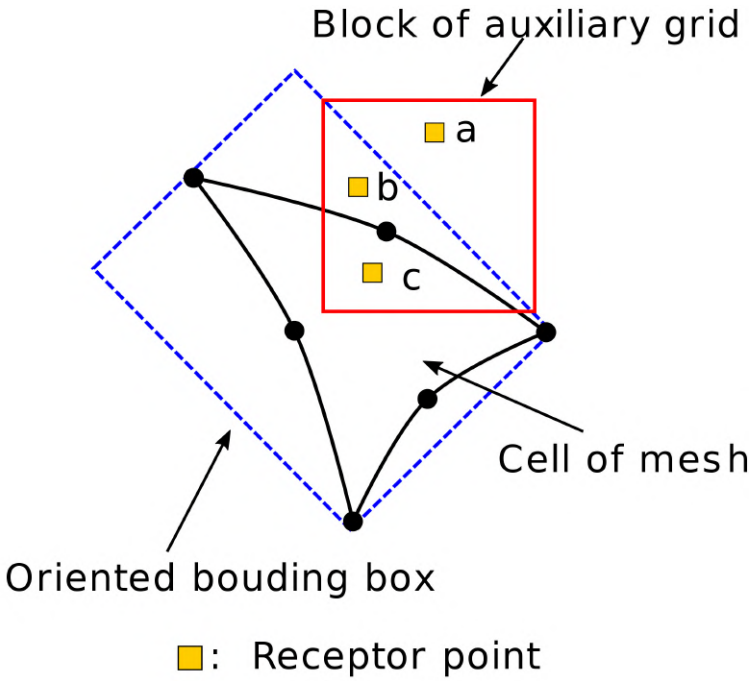



Figure 3.6: Relative positions of the receptor points to the candidate donor cell.

removed from the candidates. If the receptor point is in side the OBB, either located at the exterior (point b) or the interior (point c) of the cell, the Newton's iterations are still performed. After the donor cell is determined, the interpolated solution is calculated as

$$Q(\xi,\eta,\zeta) = \sum_j \psi_j(\xi,\eta,\zeta) Q_j, \tag{3.11}$$

where $Q_j$ is the solution value at the $j$th solution point. Similarly, the derivatives of the solutions are calculated as

$$\begin{cases} \dfrac{\partial Q}{\partial x} = \sum\limits_j \left[ \dfrac{\partial \Psi_j}{\partial \xi}\dfrac{\partial \xi}{\partial x} + \dfrac{\partial \Psi_j}{\partial \eta}\dfrac{\partial \eta}{\partial x} + \dfrac{\partial \Psi_j}{\partial \zeta}\dfrac{\partial \zeta}{\partial x} \right] Q_j, \\ \dfrac{\partial Q}{\partial y} = \sum\limits_j \left[ \dfrac{\partial \Psi_j}{\partial \xi}\dfrac{\partial \xi}{\partial y} + \dfrac{\partial \Psi_j}{\partial \eta}\dfrac{\partial \eta}{\partial y} + \dfrac{\partial \Psi_j}{\partial \zeta}\dfrac{\partial \zeta}{\partial y} \right] Q_j, \\ \dfrac{\partial Q}{\partial z} = \sum\limits_j \left[ \dfrac{\partial \Psi_j}{\partial \xi}\dfrac{\partial \xi}{\partial z} + \dfrac{\partial \Psi_j}{\partial \eta}\dfrac{\partial \eta}{\partial z} + \dfrac{\partial \Psi_j}{\partial \zeta}\dfrac{\partial \zeta}{\partial z} \right] Q_j. \end{cases} \tag{3.12}$$

The interpolated solutions and derivatives are then sent to the processes where the receptor flux points are hosted.

The steps for the parallel implement of the hole cutting and donor searching are summarized in Table 3.2.

Table 3.2: A summary of steps for the hole cutting and donor searching in parallel environment.

1. Find the local minimum and maximum of the Cartesian coordinates of the local boundary patches of the near body mesh; Communicate with all processes and determine the global extrema as well as the average boundary cell size; Build the auxiliary Cartesian grid using $(x_{min}, y_{min}, z_{min}, x_{max}, y_{max}, z_{max})$ and $(N_x, N_y, N_z)$ at each process.
2. Mark local intersected blocks and broadcast the indices of the intersected blocks to all processes, then mark global intersected blocks.
3. Use the scanline flood-fill algorithm to mark exterior blocks and interior blocks.
4. Find the cells of the background mesh that only intersect the interior blocks; Mark these cells as inactive.
5. Do collision test for the OBB of each potential interior background cell; Mark the background cell as inactive if the tags of all intersecting blocks are interior; Thus the hole boundary of the background mesh is found.
6. Build a mapping and its inverse between blocks of the auxiliary Cartesian grid and the cells of the overset meshes by using the OBB collision test.
7. For each receiver flux/solution point, use equation (3.5) to find the containing blocks; From the block-cell mapping determine the candidate donor cells;
8. Send the physical coordinates of the receptor flux/solution point to the processes where the candidate donor cells are hosted; Filter out the candidate donor cells whose OBBs do not contain the receptor point. Use equation (3.11) to determine the true donor cell; Therefore the donor cell is found and the receptor-donor connectivity is built.

## 3.4 Extension to Moving Grids

For overset meshes with moving boundaries, the overlapping region between the near-body mesh and the background mesh changes at each time step. Therefore the hole cutting and receptor-

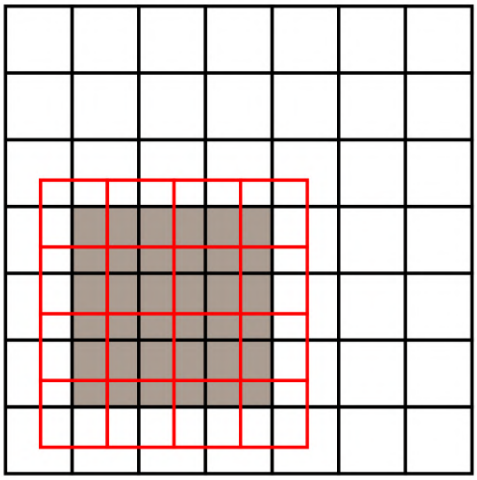
donor connectivity need to be updated at each time step. Note the hole cutting also involves deactivating and activating background cells. Some active cells at the current time step $t^n$ may become inactive at the next time step $t^{n+1}$, and vice versa. This changing of cell status is also known as blanking/unblanking in (Crabill et al., 2016b). For cells that will be deactivated at the next time step, the flow solver does not update the solutions in these cells. For current inactive cells that will be active at the next time step, interpolated solutions need to be received from the donor cells. Starting from the interpolated values, the solutions will be updated at the next time step by the flow solver. Obviously one should not wait until the next time step to perform the interpolations, because some of the currently inactive cells may already move outside the near body mesh at the next time step. The activation of these cells will fail since no donor cells can be found.

In the present work, we use a very robust approach to handle the activation/deactivation of background cells. Figure 3.7 illustrates how the activation/deactivation is performed. The red grid is the near-body mesh and the black grid is the background mesh. The gray cells are the inactive cells (or hole cells), the white cells are active cells, and the blue cells are the cells whose status are changed from inactive to active at the next time step. First, a copy of the hole-cutting information and the coordinates of the outer boundary of the body mesh is made at the current time step (Figure 3.7(a)). Then the coordinates of the outer boundary at the next time step is computed based on the body motion. Hole-cutting is performed again with the new coordinates so that two sets of inactive cells are stored (Figure 3.7(a) and (b)). A subset of cells which are inactive at $t^n$ and active at $t^{n+1}$ can be found from a simple comparison, shown as blue cells in Figure 3.7(c). This subset of cells are denoted as transition cells. The transition cells are immediately activated at $t^n$ by interpolated flow solutions from the body mesh. The flow solver also updates these cells from the current time step to the next time step. Note that the transition cells are all inactive at $t^n$. Thus it is guaranteed that they are all located inside the near-body mesh at $t^n$, such that they all find donors. Although two hole-cutting operations are involved, only one hole cutting is actually performed at each time step. This is because the hole-cutting for the current time step has already been done and stored.

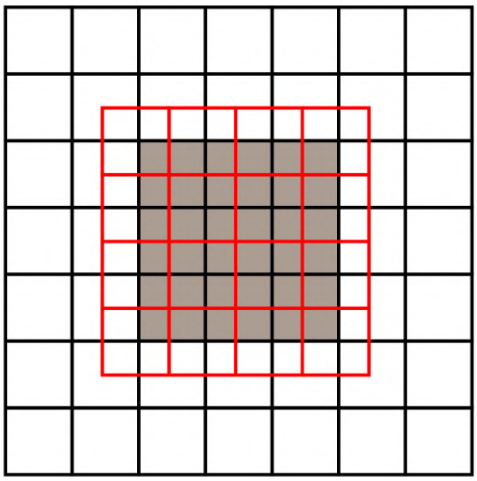
We only need to perform hole-cutting for the next time step.

## 3.5 Meshes with Sliding Interfaces

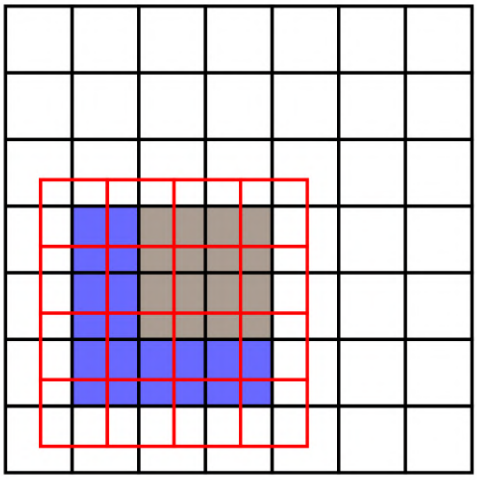
In most applications of the sliding meshes, one mesh is stationary, while the other mesh moves, e.g. rotor-stator problems. Therefore, the common interface is usually stationary. The flow solutions are communicated between the meshes sharing the common interface to achieve global coupling. The motions imposed on the the sliding interfaces are either rotation or translation. For rotations, the axis is perpendicular to the sliding interface. For translations, the motion is constraint on the sliding interface. In present work of this dissertation, we only consider translational motions to handle rotor-stator problems. The sliding meshes can be seen as a special type of moving overset meshes with no hole cutting involving. The activation/deactivation operations are therefore not needed. Similar to the overset method that we use, an auxiliary Cartesian grid is built at the sliding interface. The inverse cell-block map is still used to quickly find the candidate donor cells for each receptor point. Figure 3.8 demonstrates an example of sliding meshes in 2D space. Two triangular meshes with different resolutions contact each other at their interface. The boundaries at the sliding interface are named as sliding boundary here. Mesh 1 (red) is moving upward while Mesh 2 (blue) keeps stationary. Periodic boundary conditions are imposed along the moving direction for both meshes. Since the geometry of the sliding boundary of Mesh 1 is exactly the same with that of Mesh 2 at the beginning (3.8(a), $t = 0$), the two sliding boundaries share the same ACG (black Cartesian grid). Typically in moving overset grids the ACG needs to be updated along with the nodes' coordinates on the moving boundary of a near-body mesh. However for sliding meshes in Figure 3.8, once the ACG is built at $t = 0$, it is not updated with time. Since no motion is applied to the sliding boundary of Mesh 2, the mappings between the ACG and the cells on this sliding boundary are also not updated. The only mappings which need to be updated are between the cells on the sliding boundary of Mesh 1 and the ACG. When Mesh 1 is moving upward, the upper part of its sliding boundary is beyond the top of Mesh 2. Receptor points and donor cells on this part of the sliding boundary cannot directly find their intersecting blocks in the ACG. Instead, with the



(a) original hole cutting at $t = t^n$

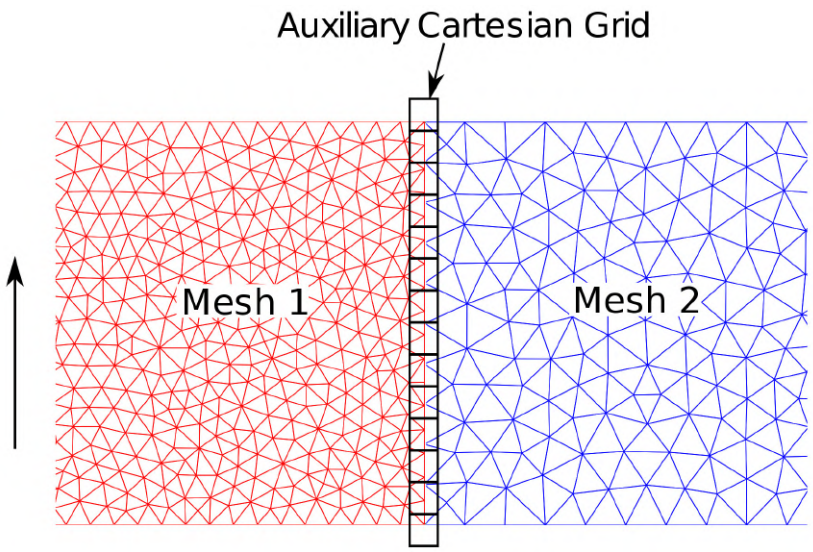


(b) original hole cutting at $t = t^{n+1}$

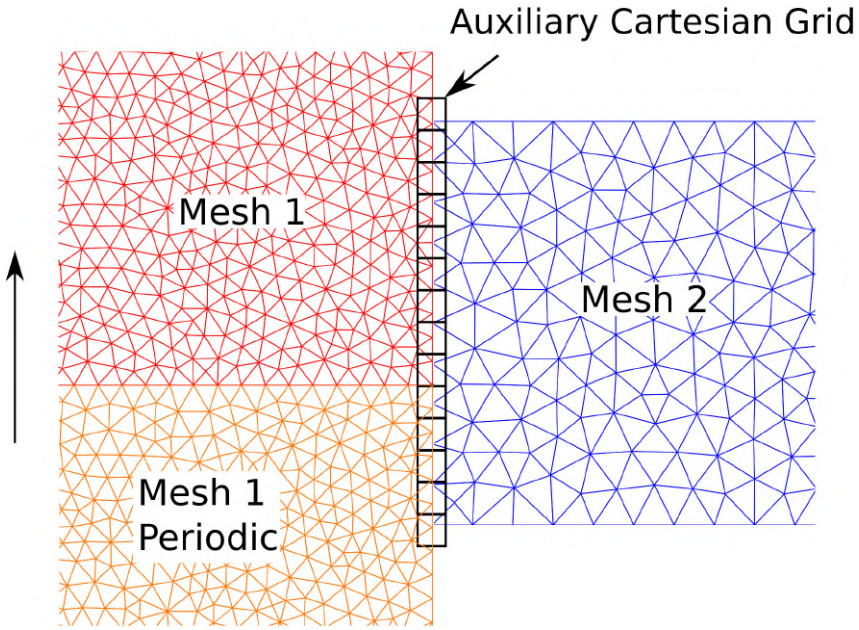


(c) modified hole cutting at $t = t^n$

Figure 3.7: Activation and deactivation of hole cells when the hole cutting is updated because of the moving near-body mesh.




(a) $t = 0$




(b) $t > 0$

Figure 3.8: Sliding meshes with a translation motion on one mesh. Periodic boundary conditions are applied on the meshes.

periodic boundary conditions, an imaginary periodic copy of Mesh 1 is generated, as shown in Figure 3.8(b). The upper part of this copy intersects with the ACG such that the mappings between the upper part of Mesh 1 and the ACG can be built through this copy of mesh. The receptor points

in Mesh 1 can now find donor cells from Mesh 2 by shifting one period. Note that when using Eq. (3.11) to compute the reference coordinates, the physical coordinates of the receptor points of Mesh 1 need to add with the amount of periodic shift. If the location of Mesh 1 moves more than one period away from Mesh 2, the periodic copy of Mesh 1 needs to be shifted by a translation of several periods until its sliding boundary intersects with the ACG.

# Chapter 4

# Time Integration Methods

Both explicit and implicit approaches are considered for unsteady flow computations. The choice depends on the flow problem and the order of accuracy. For moving grids, the solution residual not only depends on the solution, but also on the grid velocity. Therefore, the semi-discrete equation can be expressed as

$$\frac{dQ}{dt} = R(Q, \vec{v}_g). \tag{4.1}$$

We choose to compute the grid velocity in a discrete fashion so that fluid-structure interaction problems can be easily handled. In addition, we would like the time-marching approach to be a two-level one so that hole cutting and receptor-donor connectivity information is only needed for two time levels, i.e., level n and n+1. We also compute the grid velocity using the meshes at two time levels

$$\vec{v}_g^{n+1/2} = \frac{\vec{r}^{n+1} - \vec{r}^{n}}{\Delta t}. \tag{4.2}$$

As a result, the maximum time accuracy we can achieve is 2nd order (for the grid velocity) (Geuzaine et al., 2003). Fortunately, for large eddy simulations, due to the requirement of a reasonably small time step to capture the dominant turbulent dynamics, 2nd-order methods have been shown to be adequate for many applications because of the solution error is dominated by the spatial operator (Jia et al., 2019). In the following, we consider only two-level approaches with the assumption that the grid velocity is a constant, and we drop the explicit dependence on the grid velocity.

## 4.1 Runge-Kutta Methods

We consider two and three-stage explicit SSP Runge-Kutta (RK) schemes, which are 2nd and 3rd order accurate on a stationary grid. Both schemes are expected to be 2nd order with a constant grid velocity computed between time level n and n+1. The two-stage RK scheme can be written as

$$\begin{cases} Q^{(1)} = Q^n + \dfrac{1}{2} R(Q^n)\Delta t, \\ Q^{n+1} = Q^n + R(Q^{(1)})\Delta t. \end{cases} \tag{4.3}$$

The three-stage RK method is given as

$$\begin{cases} Q^{(1)} = Q^n + R(Q^n)\Delta t, \\ Q^{(2)} = \dfrac{3}{4} Q^n + \dfrac{1}{4}[Q^{(1)} + R(Q^{(1)})\Delta t], \\ Q^{n+1} = \dfrac{1}{3} Q^n + \dfrac{2}{3}[Q^{(2)} + R(Q^{(2)})\Delta t]. \end{cases} \tag{4.4}$$

We prefer the three-stage RK scheme because it allows a larger time step, and is more efficient than the two-stage counterpart. Data communications between overset grids are performed at each of the Runge-Kutta stages.

## 4.2 Backward Euler Method

For steady flow computation, we employ the backward Euler method

$$\frac{Q_i^{n+1} - Q_i^n}{\Delta t} = R_i(Q_i^{n+1}, \{Q_{nb}^{n+1}\}), \tag{4.5}$$

where $i$ is the index of the cell, $\{Q_{nb}^{n+1}\}$ is the set of all neighbor cells that contribute to the residual of cell $i$ through the normal flux jump term $[F^n]$ in Eq. (2.51). The subscript $nb$ denotes the

neighbor cells. Since the right hand side of Eq. (4.5) needs the solutions at $t^{n+1}$, it is linearized by

$$R_i(Q_i^{n+1}, \{Q_{nb}^{n+1}\}) \approx R_i(Q_i^n, \{Q_{nb}^n\}) + \frac{\partial R_i}{\partial Q_i}\Delta Q_i + \sum_{nb} \frac{\partial R_i}{\partial Q_{nb}}\Delta Q_{nb}, \tag{4.6}$$

where $\Delta Q_i = Q_i^{n+1} - Q_i^n$ and $\Delta Q_{nb} = Q_{nb}^{n+1} - Q_{nb}^n$. The $\frac{\partial R_i}{\partial Q_i}$ and $\frac{\partial R_i}{\partial Q_{nb}}$ form the diagonal and off-diagonal part of the Jacobian matrix $\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}$ respectively. Here, $\mathbf{R} = [R_1, R_2, \cdots]^T$ and $Q = [Q_1, Q_2, \cdots]^T$. The linearization transforms Eq. (4.5) into a system of linear equations. When the degree of freedoms (DoFs) are large, iterative methods (Saad, 2003) are desired to effectively solve the system of equations. However, the size of the Jacobian matrix can grow very quickly as the degree of approximate polynomials increases in high-order methods. This would require a lot of memory storage in a computer. To relief the burden of memory usage for the high-order FR/CPR method, we choose to use the block lower-upper symmetric Gauss-Seidel (LU-SGS) method (Chen & Wang, 2000; Jameson & Yoon, 1987) as the iterative solver. As shown below, the benefit of the LU-SGS method is that only the block diagonal of the Jacobian matrix needs to be stored in the memory.

Substituting Eq. (4.6) into Eq. (4.5) and rearrange, we obtain

$$\left(\frac{I}{\Delta t} - \frac{\partial R_i}{\partial Q_i}\right)\Delta Q_i = R_i(Q_i^n, \{Q_{nb}^n\}) + \sum_{nb} \frac{\partial R_i}{\partial Q_{nb}}\Delta Q_{nb}, \tag{4.7}$$

where $I$ is the identity matrix. Between $t = t^n$ and $t = t^{n+1}$, multiple forward and backward sweeps are performed until a maximum number of sweeps or the unsteady residual $\tilde{R}_i$ drops down by a given number of orders. The unsteady residual is defined as

$$\tilde{R}_i \equiv \frac{Q_i^{(k+1)} - Q_i^n}{\Delta t} - R_i(Q_i^{(k+1)}, \{Q_{nb}^*\}), \tag{4.8}$$

where $Q_i^{(k+1)}$ is the solution at the $(k+1)$th sweep, $\{Q_{nb}^*\}$ denotes the most recent solutions of the

neighbor cells. At each sweep the solutions are updated by Eq. (4.7) with the latest values:

$$\left(\frac{I}{\Delta t}-\frac{\partial R_i}{\partial Q_i}\right)\Delta Q_i^{(k+1)}=R_i(Q_i^n,\{Q_{nb}^n\})+\sum_{nb}\frac{\partial R_i}{\partial Q_{nb}}\Delta Q_{nb}^*, \tag{4.9}$$

where $\Delta Q_{nb}^*=Q_{nb}^*-Q_{nb}^n$ and $\Delta Q_i^{(k+1)}=Q_i^{(k+1)}-Q_i^n$. To keep the latest values $Q_{nb}^*$ as accurate as possible, the data communications between the overset meshes are performed at each sweep. Since

$$R_i(Q_i^n,\{Q_{nb}^n\})+\sum_{nb}\frac{\partial R_i}{\partial Q_{nb}}\Delta Q_{nb}^*\approx R_i(Q_i^n,\{Q_{nb}^*\})\approx R_i(Q_i^*,\{Q_{nb}^*\})-\frac{\partial R_i}{\partial Q_i}\Delta Q_i^*, \tag{4.10}$$

Eq. (4.9) can be transformed to

$$\left(\frac{I}{\Delta t}-\frac{\partial R_i}{\partial Q_i}\right)\Delta Q_i^{(k+1)}=R_i(Q_i^*,\{Q_{nb}^*\})-\frac{\partial R_i}{\partial Q_i}\Delta Q_i^*, \tag{4.11}$$

Adding $\frac{\partial R_i}{\partial Q_i}\Delta Q_i^*-\frac{\Delta Q_i^*}{\Delta t}$ to both sides of Eq. (4.11), a very compact form of the block LU-SGS form is obtained:

$$\left(\frac{I}{\Delta t}-\frac{\partial R_i}{\partial Q_i}\right)\Delta^* Q_i^{(k+1)}=R_i(Q_i^*,\{Q_{nb}^*\})-\frac{\Delta Q_i^*}{\Delta t}, \tag{4.12}$$

where $\Delta^* Q_i^{(k+1)}=\Delta Q_i^{(k+1)}-\Delta Q_i^*$. In order to mimic a symmetric GS (SGS) approach, Eq. (4.12) is solved in forward and backward sweeps according to the natural cell ordering in the mesh until the norm of the unsteady residual is reduced sufficiently, normally by two orders of magnitude. On each element, the linear equation is small, and is solved with a direct LU-decomposition algorithm. Therefore, the overall solution method is called the LU-SGS scheme. The matrix on the left hand side of above equation only involves the block diagonal part $[\frac{\partial R_i}{\partial Q_i}]$ of the Jacobian matrix. Therefore, comparing to other implicit methods which require the complete Jacobian matrix, this form of the LU-SGS method saves a lot of memory for the storage of the Jacobian matrix for high order methods.

## 4.3 Crank-Nicolson Method

The backward Euler scheme is often chosen due to its simplicity and stability. However the backward Euler scheme is a first-order scheme, which is not accurate enough for unsteady flow problems. The only two-level scheme which is 2nd order accurate is the Crank-Nicolson method. The Crank-Nicolson formula for element $i$ is

$$\frac{Q_i^{n+1}-Q_i^n}{\Delta t}=\frac{1}{2}[R_i(Q_i^{n+1},\{Q_{nb}^{n+1}\})+R_i(Q_i^n,\{Q_{nb}^n\})]. \tag{4.13}$$

Define the unsteady residual as

$$\tilde{R}_i(Q^{n+1})=-\frac{Q_i^{n+1}-Q_i^n}{\Delta t}+\frac{1}{2}[R_i(Q_i^{n+1},\{Q_{nb}^{n+1}\})+R_i(Q_i^n,\{Q_{nb}^n\})]. \tag{4.14}$$

An approximate Newton scheme after dropping the off-diagonal blocks to solve this equation is

$$\left(-\frac{I}{\Delta t}+\frac{1}{2}\frac{\partial R_i}{\partial Q_i}\right)\Delta Q_i=-\tilde{R}_i(Q^n). \tag{4.15}$$

With a Newton solver, an outer loop is necessary to converge the unsteady residual to a user-specified tolerance. Instead, we employ the following non-linear Gauss-Seidel (GS) like approach to directly drive the unsteady residual to convergence

$$\left(-\frac{I}{\Delta t}+\frac{1}{2}\frac{\partial R_i}{\partial Q_i}\right)\Delta Q_i^{(k+1)}=-\tilde{R}_i(Q^*), \tag{4.16}$$

where $k$ is an iteration index, and $Q^*$ is the latest available solution in the iterative process. The LU-SGS solver for the Crank-Nicolson scheme has two advantages: (1) Only the block diagonal matrix on each element is stored and the memory requirement is much smaller than a fully implicit scheme; (2) Only one loop is necessary to converge the non-linear unsteady residual.

To verify the Crank-Nicolson time scheme for the overset meshes, an accuracy study with an

isentropic vortex problem is performed. The analytical solution can be found at Section 5.1. The density errors with various time-step sizes are shown in Figure 4.1. Second order of accuracy in time is achieved for all simulation cases.

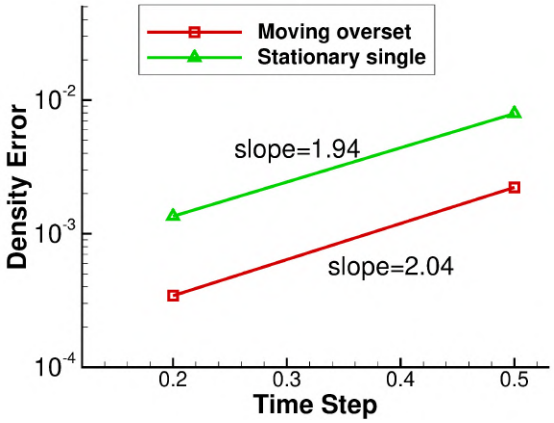



Figure 4.1: Density errors of an isentropic vortex problem on moving overset meshes and a stationary single mesh.

# Chapter 5

# Numerical Results

In this chapter, several simulation cases are run to verify and validate the high-order FR/CPR overset method for both stationary and moving/sliding grids. Section 5.1 compares the face-based data communication approach with the element-based approach. The results of accuracy studies are given in Section 5.2. Section 5.3 demonstrates a steady flow case of a sphere near-body mesh overlapping with a background mesh. Convergent flow solutions are obtained. In Section 5.4, transitional flow over a T106 turbine blade is simulated on stationary overset meshes. The results agrees well with those from a benchmark single-zone mesh case. Section 5.5 extends the T106A case to sliding meshes. Vortices are generated from moving cylinders and hit on the T106 blade after propagating across a sliding interface. The impacts of the cylinder vortices are analyzed. For moving overset meshes, a heaving and pitching NACA0012 airfoil case is run in Section 5.6. Finally, simulation results of a single-bladed rotor are shown in Section 5.7

## 5.1 Comparison of Data Communication Approaches

In order to evaluate the accuracy and stability of the face-based and the element-based interpolation approaches, simulations of an isentropic vortex in inviscid flow are performed. The 2D vortex problem has an analytical solution which was given in (Yee et al., 1999). For free-stream conditions $(\rho, u, v, p) = (\rho_\infty = 1, u_\infty, v_\infty, p_\infty = 1)$, $T_\infty = p_\infty/\rho_\infty = 1$, and no perturbation in entropy ($\delta S = 0$),

the perturbation values are

$$\begin{cases} (\delta u, \delta v) = \dfrac{\beta}{2\pi} e^{(1-r^2)/2}(-y+y_c, x-x_c) \\ \delta T = -\dfrac{(\gamma-1)\beta^2}{8\gamma\pi^2} e^{1-r^2} \end{cases}, \tag{5.1}$$

where $\beta$ is the vortex strength, $\gamma = 1.4$, $(x_c, y_c)$ are coordinates of the vortex center, and $r^2 = (x-x_c)^2 + (y-y_c)^2$. The vortex center moves with the free-stream velocity $(u_\infty, v_\infty)$, i.e. $(x_c, y_c) = (x_0 + u_\infty t, y_0 + v_\infty t)$, where $(x_0, y_0)$ is the initial vortex center. The temperature, density and pressure are given by

$$\begin{cases} T = T_\infty - \delta T = 1 - \dfrac{(\gamma-1)\beta^2}{8\gamma\pi^2} e^{1-r^2} \\ \rho = T^{\frac{1}{\gamma-1}} \\ p = \rho T = \rho^\gamma \end{cases}. \tag{5.2}$$

This case was used to study the accuracy and stability of high-order methods in long term simulations (Spiegel et al., 2015), and also is chosen here to compare the stability of the two different data communication approaches.

Figure 5.1 shows the overset meshes and the initial condition for the present simulation. The computational domain is $[-5,5] \times [-5,5]$ on the XY plane, and $[-2.5, 2.5]$ in the Z direction. The initial vortex center is $(0,0)$ on the XY plane. The background mesh has a dimension of $40 \times 40 \times 10$, while the dimension of the near body mesh is $15 \times 15 \times 10$ such that it has a similar resolution as the background mesh. Two different sets of free-stream conditions are tested for the vortex problem. In the first test, the free-stream conditions are $(\rho_\infty, u_\infty, v_\infty, w_\infty, p_\infty) = (1,1,1,0,1)$. The vortex has a convective motion inside the XY plane with a velocity of $(1,1)$. In the second set the free-steam conditions are $(\rho_\infty, u_\infty, v_\infty, w_\infty, p_\infty) = (1,0,0,0,1)$, such that the vortex is stationary. Periodic boundary conditions are imposed on all boundaries of the background mesh for all simulations. For the convective vortex, a maximum of 10 periods were simulated unless the simulation crashed earlier. Since the domain size is 10 in both the X and Y directions, one period of the simulation corresponds to 10 time units. For the stationary vortex, a maximum of 70 time units

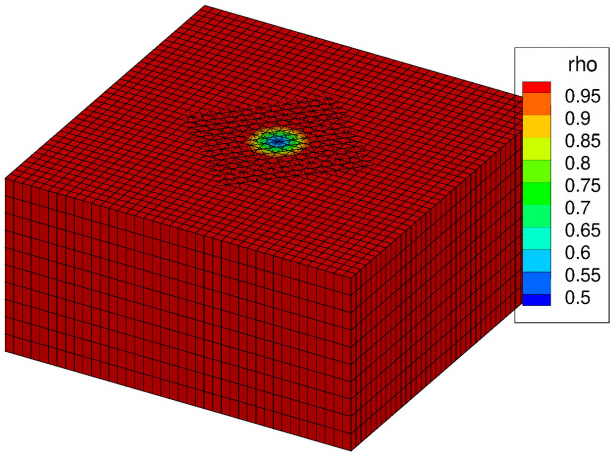
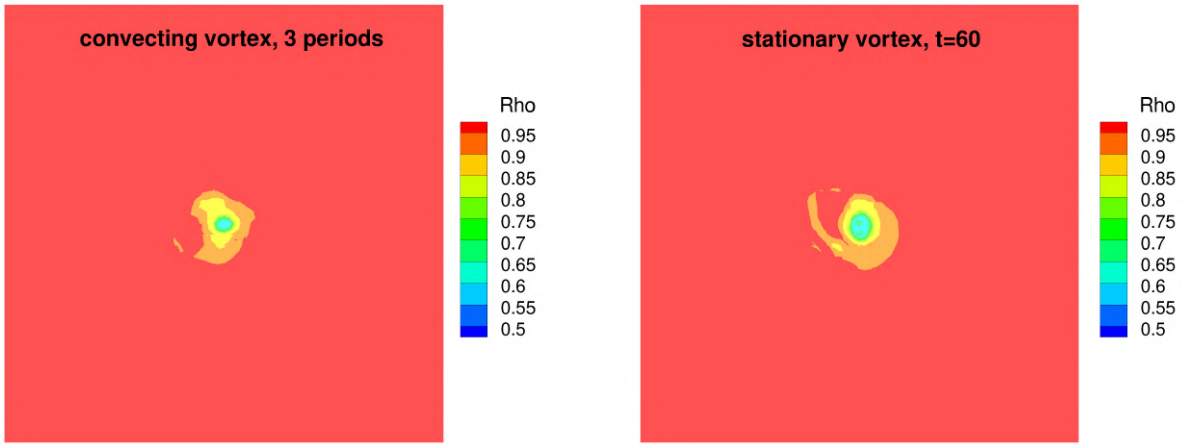



Figure 5.1: The overset meshes and the initial flow field to compare face-based interpolation and element-based interpolation approaches.

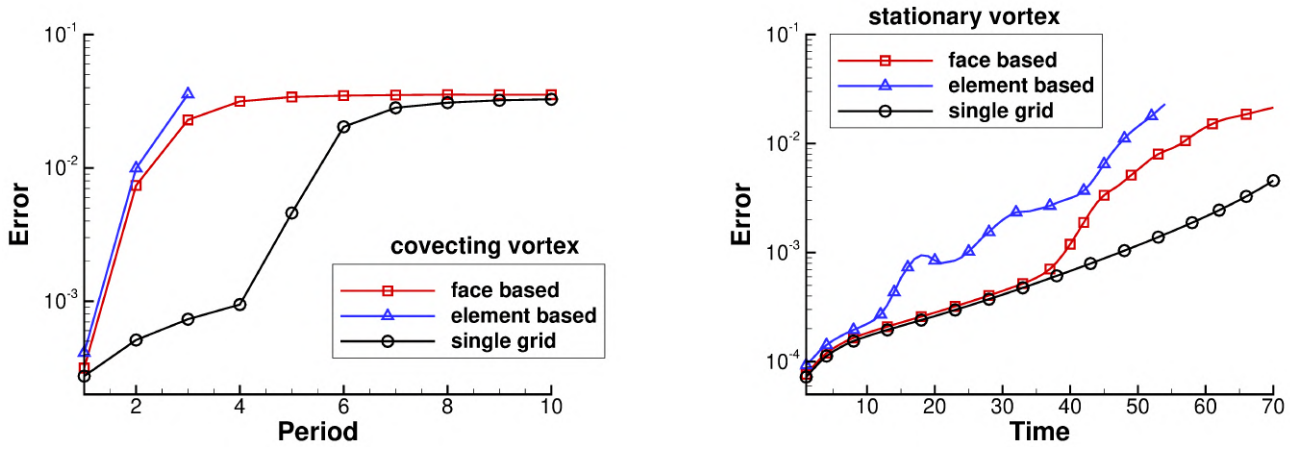
were simulated. For comparison purposes, the same simulations were also performed on a single mesh. All the simulations were run with a degree 2 solution polynomial, i.e. $p = 2$. Figure 5.2 shows the density contours on the XY plane with overset meshes. For the convective vortex, the vortex center moves back to its initial location after each period. In both convective and stationary vortex cases, the solution is still stable at $t = 10$ (or 1 period). However, due to the accumulation of numerical errors, the solution has already become unstable after $t = 30$ (or 3 periods). Figure 5.3 shows that the $L_2$ norm of the density error for both the overset and the single meshes with the two interpolation approaches. For the convective vortex case, the simulation with the element-based interpolation approach crashed in only 3 periods (red curve in Figure 5.3(a)). In contrast, the simulation with the face-based interpolation approach continued until the maximum number of periods (10 periods) was reached (blue curve in Figure 5.3(a)), although the error grew fast from the beginning and reached the maximum after 4 periods. For the simulation on the single mesh, the density error remained low until 4 periods and reached the maximum after 8 periods (black curve in Figure 5.3(a)). For the stationary vortex case, again the simulation with the element-based interpolation approach showed an instability and crashed at $t = 55$ (red curve in Figure 5.3(b)), while the simulation with the face-based interpolation approach lasted to the maximum simulation time $t = 70$ (blue curve in Figure 5.3(b)). The error from the face-based approach is very close to

that from the single-mesh simulation (black curve in Figure 5.3(b)) until about $t = 40$, while the error from the element-based approach departed from the single-mesh result as early as $t = 12$. It is clearly seen from Figure 5.3 that the face-based interpolation approach is not only more stable but also more accurate than the element-based interpolation approach. Therefore in the rest cases of this dissertation, only the face-based approach is used.




(a) convecting vortex (b) stationary vortex

Figure 5.2: Density contours of an isentropic vortex at t=60 or 3 periods.




(a) convecting vortex (b) stationary vortex

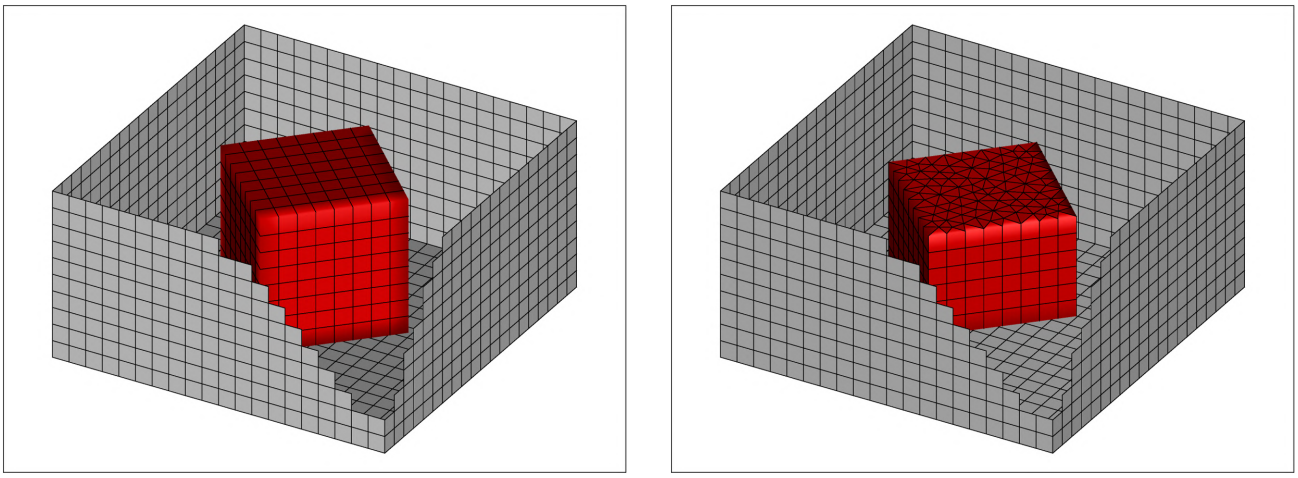
Figure 5.3: Density errors of isentropic vortex cases with different interpolation approaches.

## 5.2 Accuracy Studies

Accuracy study is an import step to verify the implement of a high-order method. In this section, accuracy studies are performed on stationary/moving overset meshes and sliding meshes. For inviscid flow, the convecting vortex case is used. For viscous flow, the Couette problem is solved.

### 5.2.1 Stationary Overset Meshes

We firstly perform the accuracy studies on stationary overset meshes for both inviscid and viscous flows. Two sets of overset grid systems are generated for the simulations. One overset system consists two hexahedral meshes. The mesh sizes of the background meshes in this set are $10 \times 10 \times 10$, $20 \times 20 \times 10$ and $40 \times 40 \times 20$. The second overset system is generated from a prism near-body grid and a hexahedral background grid. The domain is $[-5,5] \times [-5,5]$ on the XY plane, and $[-2.5,2.5]$ in the Z direction. Overset meshes with mesh size $20 \times 20 \times 10$ are shown in Figure 5.4 as an example.

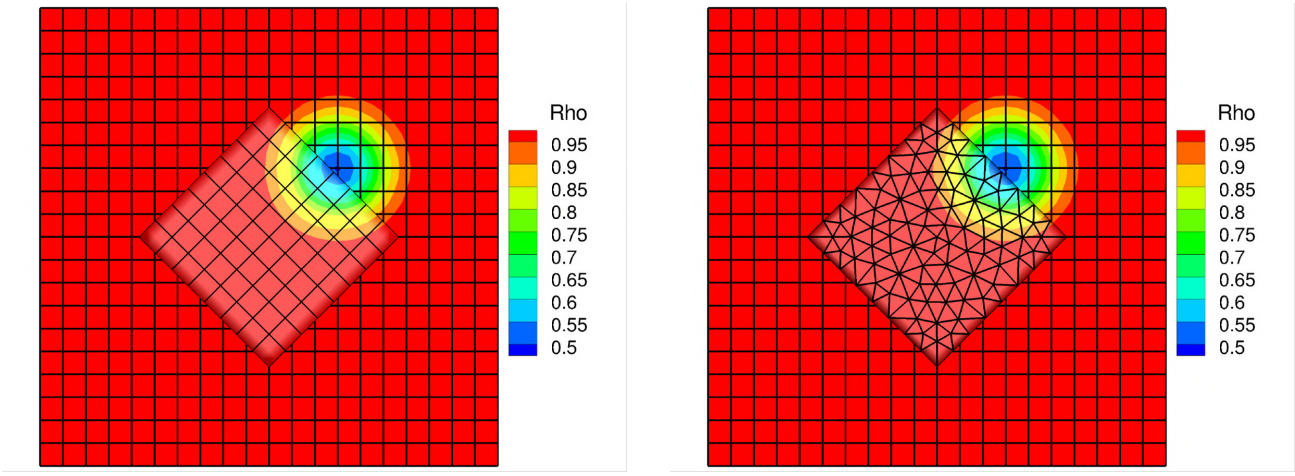


(a) hexahedral background mesh and hexahedral near-body mesh

(b) hexahedral background mesh and prism near-body mesh

Figure 5.4: Overset meshes with background grid size $20 \times 20 \times 10$.

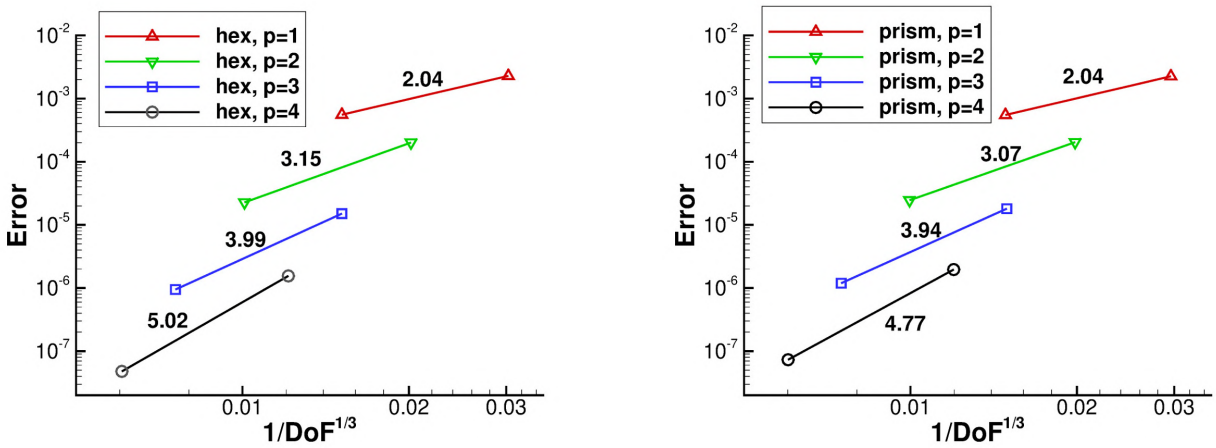
The convecting isentropic vortex with analytical solutions given by Eq. (5.1) and (5.2) is again simulated on those meshes. The initial conditions are similar to those in Figure 5.1. The boundary




(a) hexahedral near-body mesh

(b) prism near-body mesh

Figure 5.5: Density contours of the insentropic vortex on the XY plane. Simulations run on stationary overset meshes



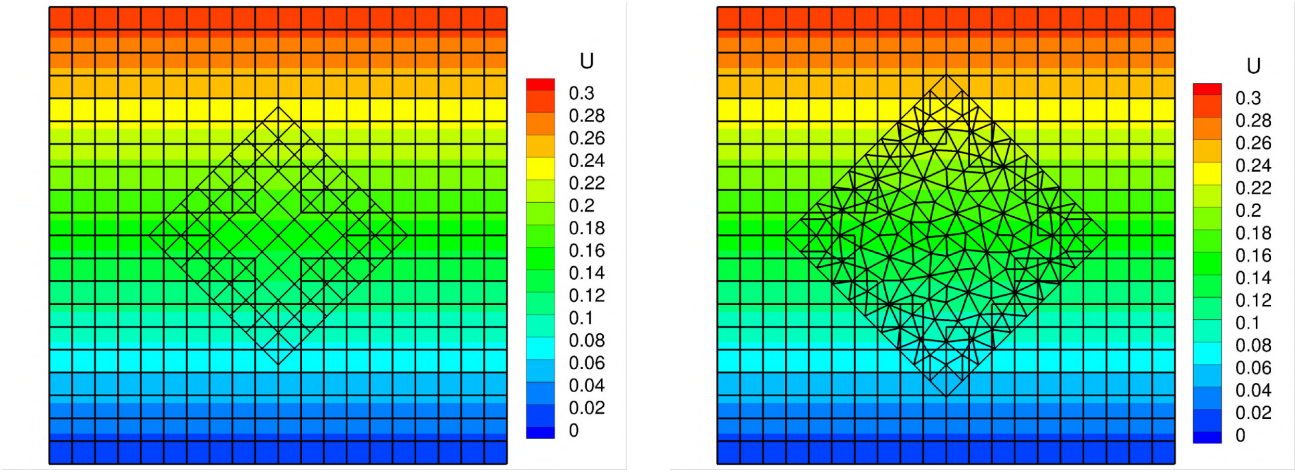

(a) hexahedral near-body mesh

(b) prism near-body mesh

Figure 5.6: $L_2$ error for the convecting isentropic vortex on stationary overset grids for various grid sizes and polynomial degrees. The numbers besides the line segments are the slopes.

conditions on all the six faces of the background mesh are set to the analytical solution. The degree of the approximate solution polynomials varies from $p = 1$ through $p = 4$. Each simulation runs for 1.5 time units. The density contours on the XY plane at $t = 1.5$ are shown in Figure 5.5, with the size of the background mesh $20 \times 20 \times 10$. At $t = 1.5$ the vortex center is located at the boundary of the near-body mesh. Figure 5.6 shows the $L_2$ error of density for both hexahedral and prism overset grids with polynomial degrees ranging from $p = 1$ to $p = 4$. For a finite element

based method the errors should decrease as $DOF^{-(p+1)/d}$, where $d$ is the dimension of space. The slopes in Figure 5.6 are close to the expected values, which demonstrates that the overset FR/CPR method preserves the formal order of accuracy.



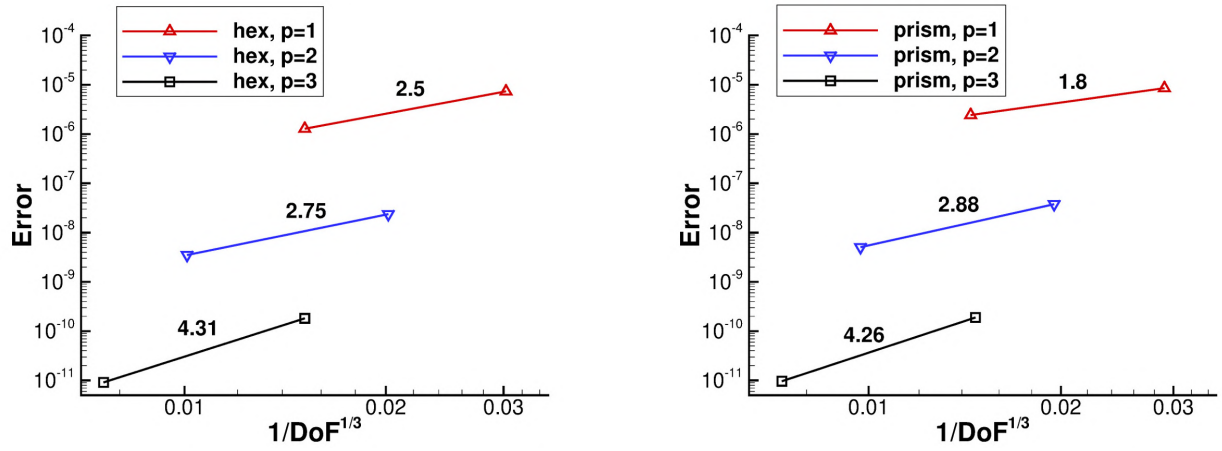

(a) hexahedral near-body mesh

(b) prism near-body mesh

Figure 5.7: Steady $u$ velocity distributions of the Couette flow on the XY plane. The simulations are run on stationary overset meshes.

To demonstrate the solution accuracy for viscous flows, the Couette problem is simulated on stationary overset meshes with different mesh resolutions, similar to those used for the isentropic vortex case. However, the domain is changed to $[0,10]$ on all spatial dimensions. The Couette problem describes a compressible viscous flow between two parallel plates, with one moving and one stationary. This problem has a steady analytical solution. Assume that the distance between the two plates is $L$. The top plate moves along the x direction with a speed of $U$ while the bottom plate is stationary. Subscript $b$ denotes the flow properties of the bottom plate and $t$ for the top plate, with temperature $T_b$ and $T_t$ given. Neglecting the pressure gradient, the steady analytical solutions are

$$\begin{cases} T = T_b + \frac{z}{L}(T_t - T_b) + \frac{P_r}{C_p}U^2\frac{z}{2}(1-\frac{z}{L}) \\ \rho = \frac{p}{\gamma T} \\ u(z) = U\frac{z}{L} \end{cases}, \tag{5.3}$$




(a) hexahedral near-body mesh

(b) prism near-body mesh

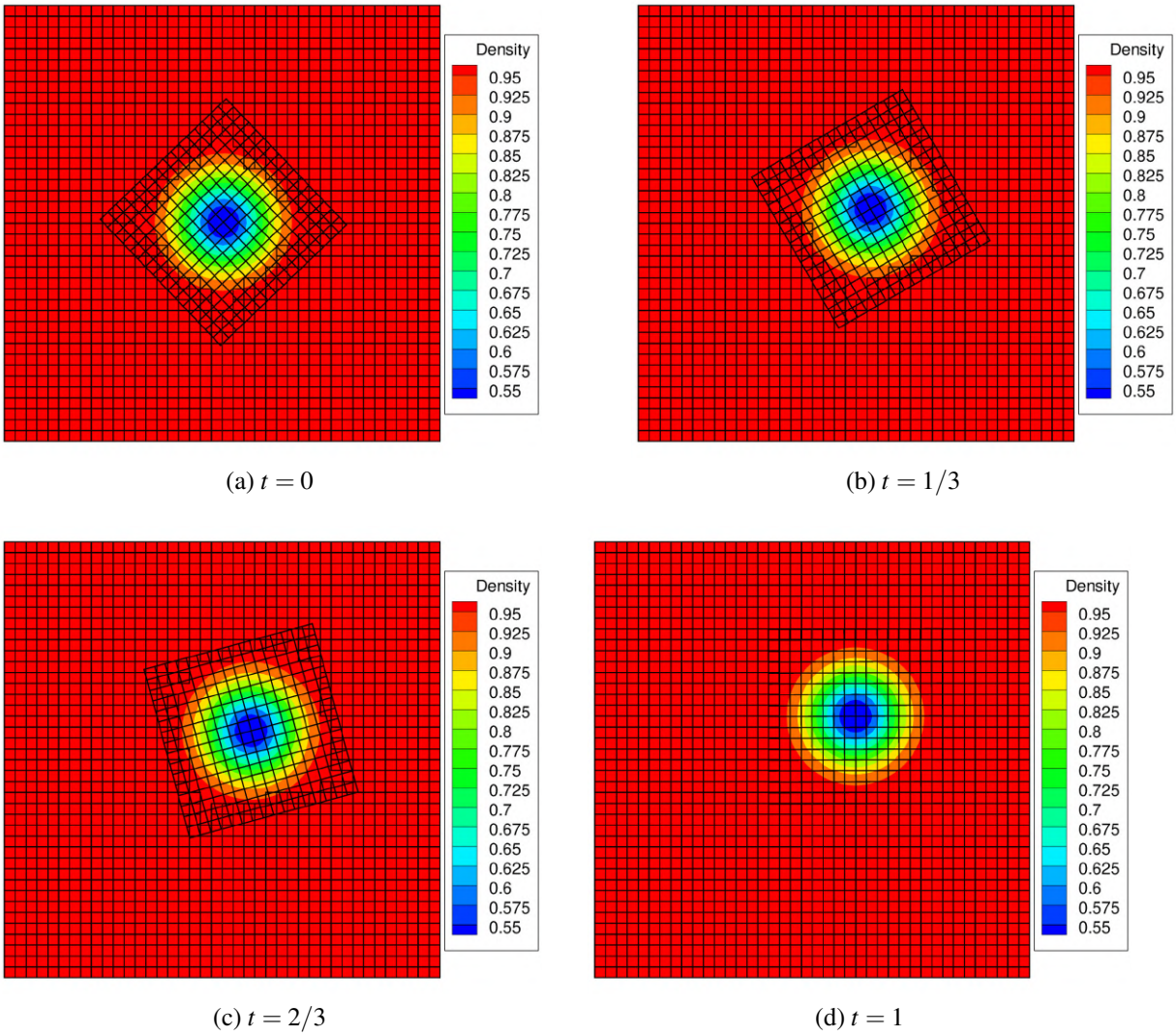
Figure 5.8: Errors of the $u$ velocity of the Couette flow. The numbers are the slopes of the segments. The simulations are run on stationary overset meshes.

where $P_r$ is the Prandtl number, $C_p$ is the specific heat, and $\gamma = 1.4$. The x-velocity contours of the steady state solutions are shown in Figure 5.7. The $u$ velocity shows a linear distribution along the normal direction ($y$ axis), which is consistent with the analytical solution $u(y) = U\frac{y}{L}$. The $L_2$ errors of the $u(y)$ are shown in Figure 5.8. The slopes of the line segments show that the FR/CPR method on overset meshes achieve the formal order of accuracy for viscous flows.

### 5.2.2 Moving Overset Meshes

We then extend the hp-refinement study to moving overset meshes. For the isentropic vortex case, two sets of overset meshes with different resolutions are used for the hp-refinement study. The dimension of the background Cartesian mesh is $20 \times 20 \times 10$ for the coarse mesh, and $40 \times 40 \times 20$ for the fine mesh. To test the overset interface treatment, another mesh containing the convecting vortex is also generated, and behaves like a "body" mesh to overlap the background Cartesian mesh. The "body" meshes have the same resolution as the background meshes. The vortex moves on the x-y plane with the velocity of (1, 1). The body mesh moves as a rigid body with a translational velocity (1, 1), and an angular velocity $\omega = 0.25\pi$ with respect to the geometric center of the body mesh. The translation is designed so that the center of the body mesh always coincides with the

center of the vortex. Density contours at $t = 0$, $t = 1/3$, $t = 2/3$, and $t = 1$ with a p2 simulation on

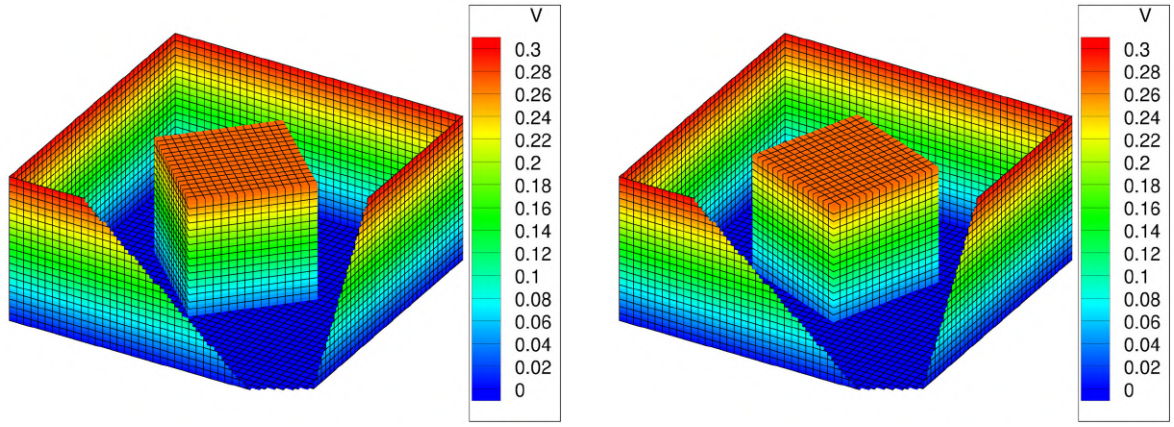



(a) $t = 0$ (b) $t = 1/3$

(c) $t = 2/3$ (d) $t = 1$

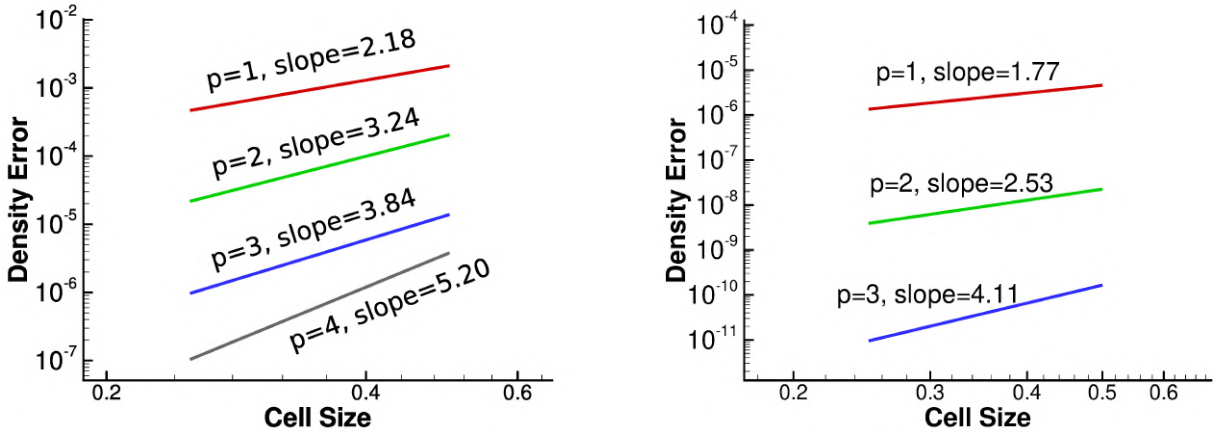
Figure 5.9: Snapshots of convecting vortex on moving overset grids.

the fine overset meshes are shown in Figure 5.9 as an example. The body mesh moves and rotates together with the vortex. The Couette flow is simulated on the same sets of overset meshes. A spinning motion at a constant angular velocity $\omega = 0.25\pi$ with respect to the z axis is imposed on the "body" mesh. No translation is involved in the Couette flow case. Snapshots of the steady-state flow velocity field at different simulation times are shown in Figure 5.10. Note that the velocity is linear. The errors for p1 to p4 simulations for both the vortex case and the Couette flow case are given in Figure 5.11. The designed order of accuracy is preserved.




(a) $t = 80$ (b) $t = 81$

Figure 5.10: Snapshots of the velocity field of a Couette flow at different simulation time. The body mesh is spinning in the background mesh.

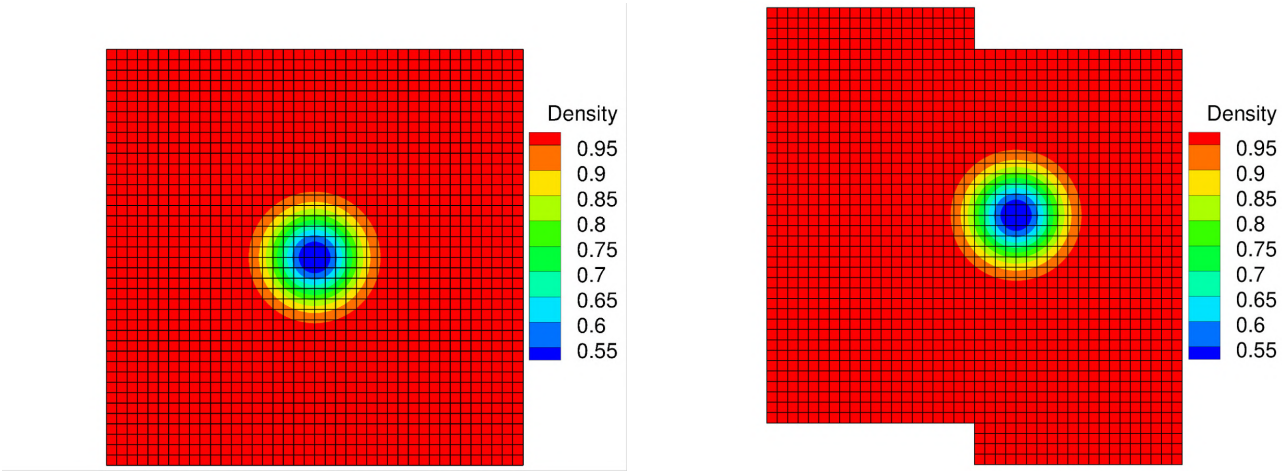



(a) vortex case (b) Couette flow case

Figure 5.11: Density errors of convecting vortex problem and Couette flow problem on moving overset meshes.

### 5.2.3 Sliding Meshes

We use the vortex case again to perform an accuracy study for the FR/CPR method with sliding meshes. Figure 5.12 shows the density fields of the vortex moving in inviscid flow. The simulation is performed on two adjacent meshes with a sliding interface. The mesh on the left side moves upward with a constant velocity $v_g = 1$ while the mesh on the right side keeps stationary. The size of each sliding mesh is $20 \times 40$. The initial position of the vortex center locates on the sliding




(a) $t = 0$ (b) $t = 1$

Figure 5.12: Density errors of a convecting vortex at different simulation time on sliding meshes.

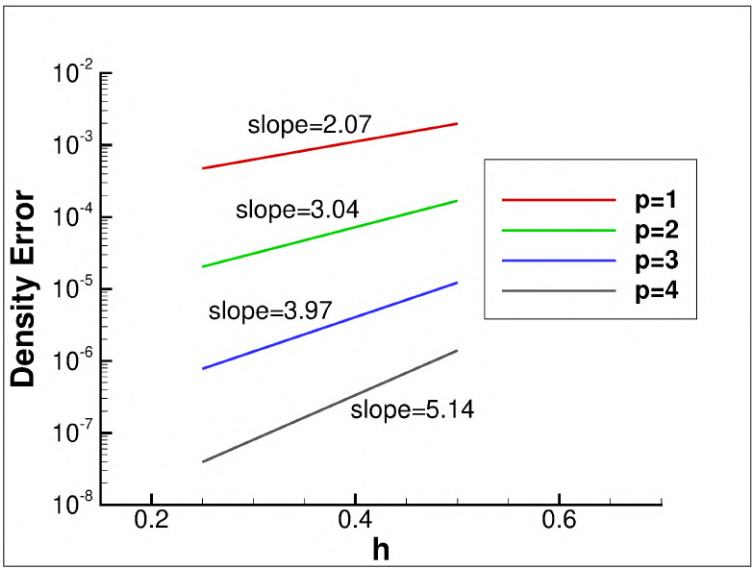
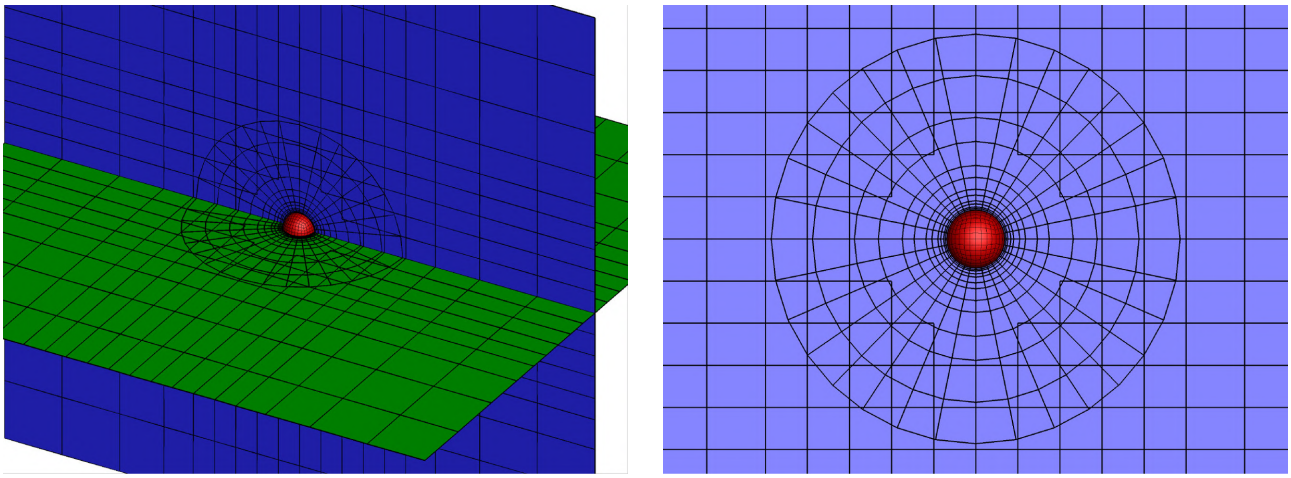



Figure 5.13: Density errors of the simulations of a convecting vortex on sliding meshes.

interface. The vortex moves in XY plane with a velocity of $(1,1)$. The entire vortex moves into the right side stationary mesh at the end of the simulation ($t = 1$). The same simulation is run on coarser sliding meshes with a mesh size of $10 \times 20$ for each side. The density errors for degree 1 through degree 4 polynomial approximations are given in Figure 5.13. The slopes show the designed order of accuracy is achieved.

## 5.3 Steady Flow Over a Sphere

This case demonstrates the performance of the implicit solver for solving steady flow problems on overset grids. The flow is nearly incompressible with a Reynolds number of 118 and Mach number of 0.2535. Figure 5.14 shows the overset grids of a near-body mesh of a sphere and a Cartesian background mesh. To represent the spherical geometry better, the linear elements of the near-body mesh are upgraded to quadratic elements by using MeshCurve (Ims et al., 2015). The outer boundary of the near-body mesh is 3.5 diameters away from the center of the sphere, while the outer boundary of the background grid is 10 diameters away.



(a) overall looking (b) zoom in to near-body mesh

Figure 5.14: The overset meshes of a near-body mesh with spherical wall boundary and a Cartesian background mesh.

The block LU-SGS method is used with the backward Euler scheme. Density residual histories are shown in Figure 5.15 for $p = 1$ through $p = 3$ simulations. For all simulations, the residual drops at least 10 orders of magnitude. The steady Mach number contours with $p = 3$ are shown in Figure 5.16. The streamlines from the simulations are also compared to the streamlines from an experiment (Taneda, 1956) in Figure 5.17. Note that there is good qualitative agreement on the size of and location of the separation bubble between the simulation and experiment. The size of the separation region in the simulation, which is 1.04 diameter of the sphere, agrees well with that of experiment.

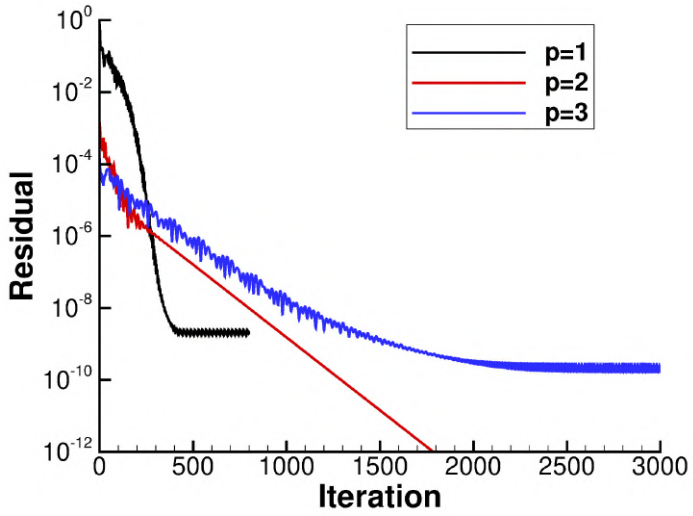



Figure 5.15: Convergence histories of flow over a sphere using different degrees of solution polynomials ($p = 1$ through $p = 3$).

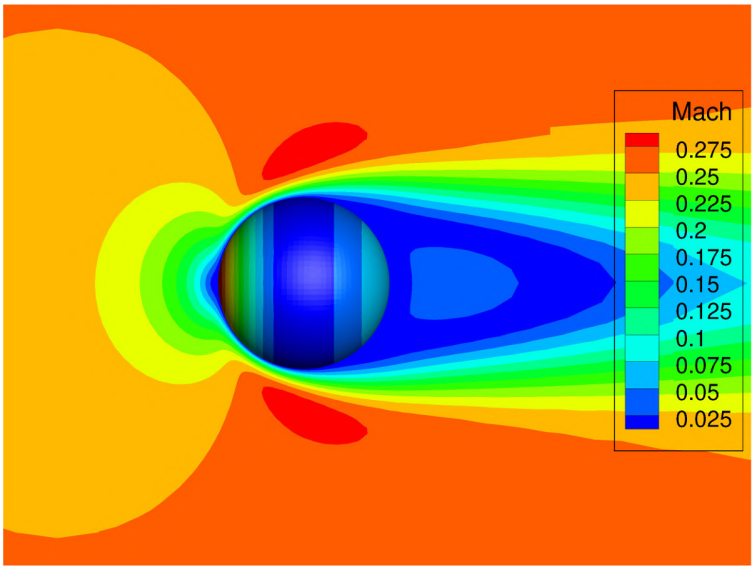
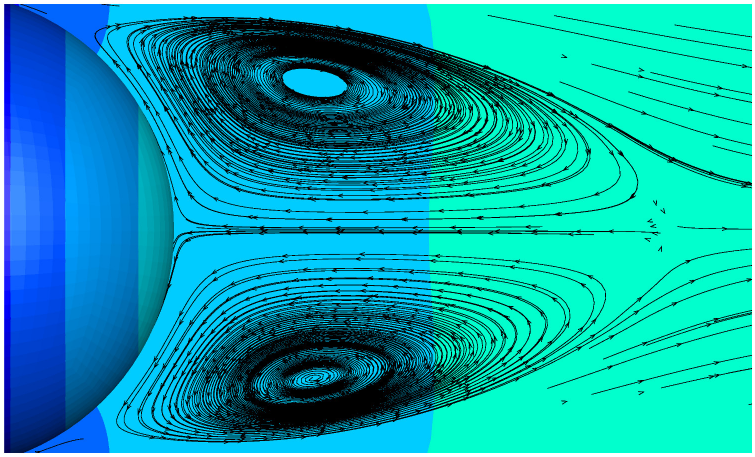
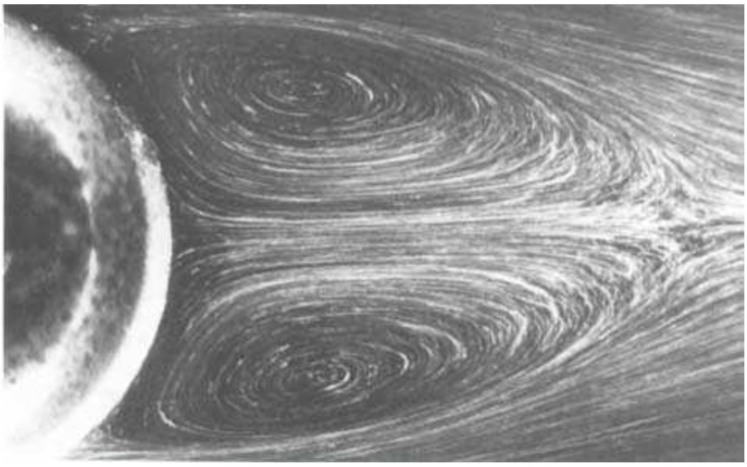



Figure 5.16: Mach number (on z=0 plane) and pressure (on the sphere) contours from the 4th order overset FR/CPR simulation.

Figure 5.18 shows the skin friction coefficient profiles on the sphere wall between the current simulations, and another one from the literature using a 6th order spectral difference scheme (Sun et al., 2007). The angle is defined 0 at the stagnation point on the wind-side. For $p = 1$, the profile differs slightly from that of the 6th order SD. For $p = 2$ and $p = 3$, the profiles are on top of that of the 6th order SD. This indicates that the skin friction coefficient has reached p-convergence since



(a) simulation



(b) experiment

Figure 5.17: Comparison of streamlines between simulation and experiment.

$p = 2$ for overset meshes. According to the converged skin friction profiles, the separation angle is 123.6 degrees, which also agrees well with result from the literature.

## 5.4 Transitional Flow over the T106A Turbine Blade

A benchmark unsteady transitional flow problem is chosen to demonstrate the high-order FR/CPR overset solver for turbulent flow problems. This case is from the 4th International Workshop on High-Order CFD Methods (`https://how4.cenaero.be`). The single p2 mesh shown in Figure 5.19(b) is provided by the workshop. The mesh has 5 layers of elements in the spanwise direction and the span is 10% of the chord length. Overset grids of the T106A blade (Figure 5.19(a)) are

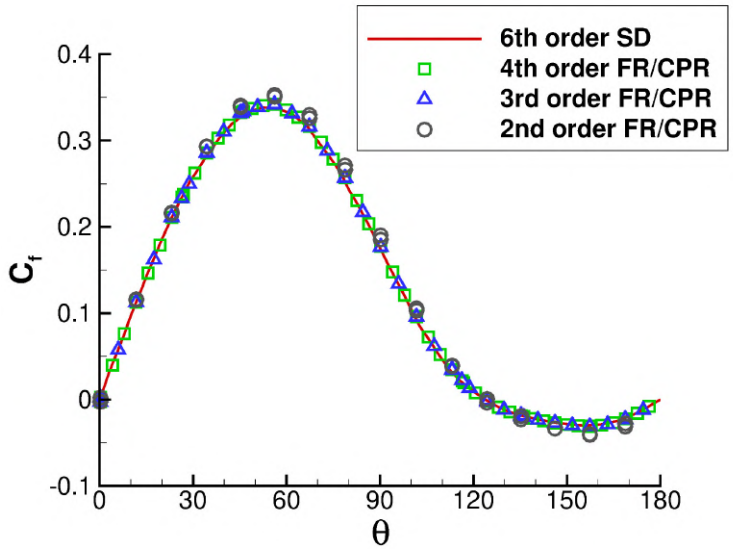
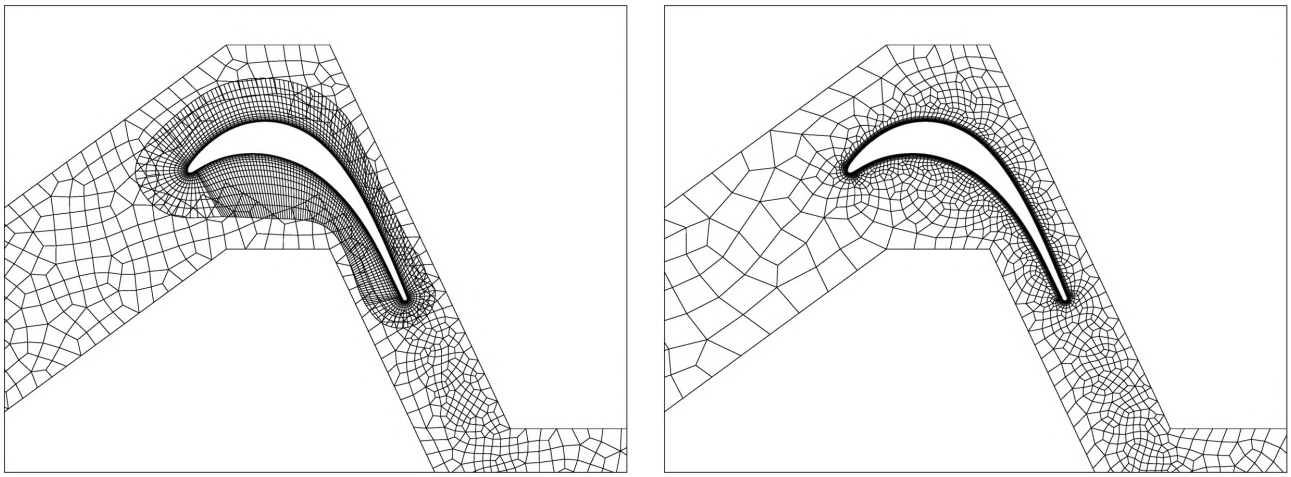



Figure 5.18: Computed skin friction coefficient profiles with $p = 1$ through $p = 3$ overset FR/CPR methods. $\theta$ is the angle to the wind side stagnation point with respect to the center of the sphere. The profile from a 6th order spectral difference (SD) method on a single mesh serves as a benchmark.

generated with a similar resolution at the wall surface as the single mesh. The angle of the



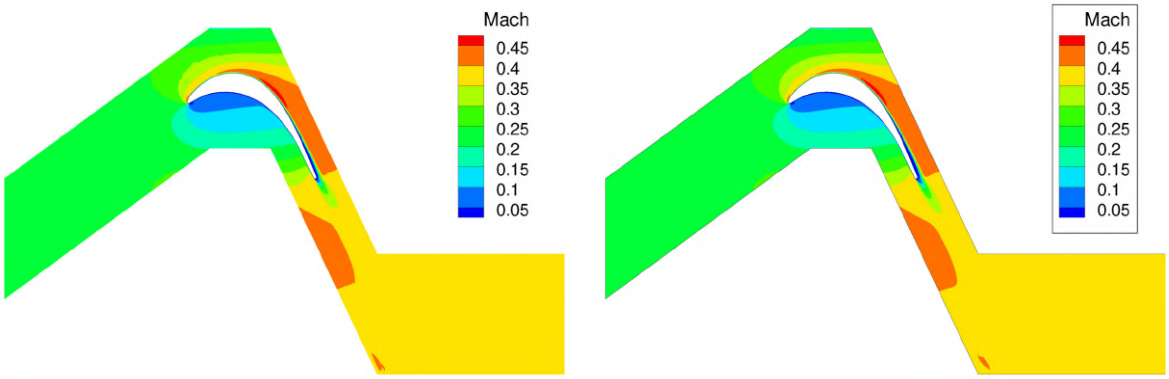
(a) (b)

Figure 5.19: High-order meshes of the T106A turbine blade.

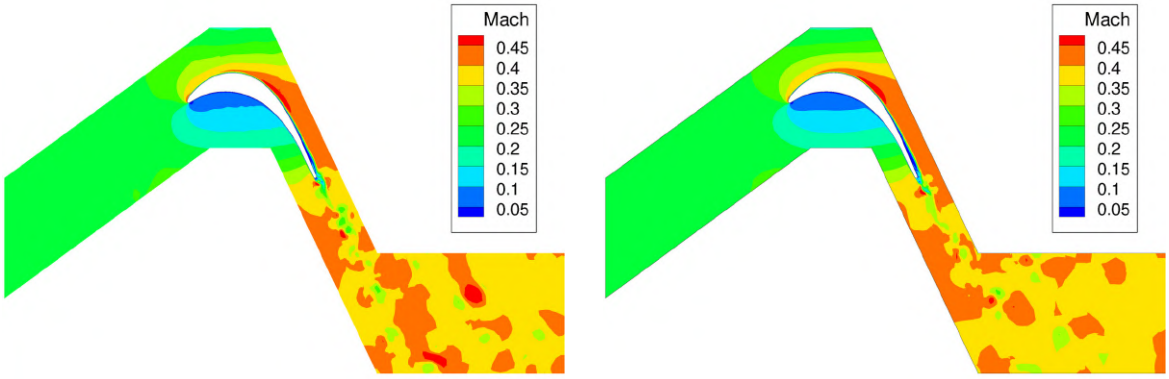
incoming flow with respect to the x-axis is 46.1°. The isentropic Mach number is 0.4 and the Reynolds number is 60,000 based on the chord length and the insentropic exit condition. The average $y+$ at the wall cells is about 4.9. Periodic boundary conditions are applied on both the

(a) overset grids (b) single grid

Figure 5.20: Iso-surface of Q-criterion colored by spanwise vorticity for both the overset mesh and the single mesh.



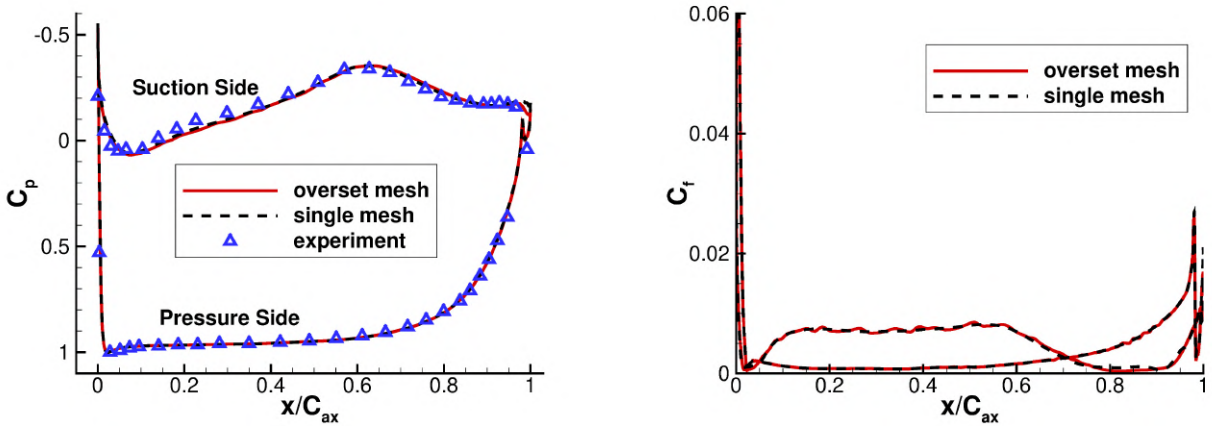

(a) mean flow, overset grids (b) mean flow, single grid




(c) instantaneous flow, overset grids (d) instantaneous flow, single grid

Figure 5.21: Mean and instantaneous Mach number contours for the T106A turbine blade.

spanwise and pitchwise directions. Simulations are run on both the overset meshes and the single mesh for comparison at $p = 2$. The time integration scheme for all simulations is RK3. Mean flow solutions, instantaneous flow solutions, the mean surface pressure coefficient $c_p$, and the mean skin friction coefficient $c_f$, are compared between the overset and the single mesh results. The averaging of the flow solutions begins at a time when the transient solution passes through the whole domain and the mean flow is sufficiently converged. Figure 5.20 shows the instantaneous iso-surfaces of the Q-criterion colored by the magnitude of vorticity in the spanwise direction. The laminar flow breaks down and transitions into turbulence at a location close to the trailing edge for both the overset and the single mesh simulations. Figure 5.21 shows the mean and instantaneous Mach number fields for both grids. As expected, the mean-flow Mach number contours from the overset simulation are nearly identical to those from the single mesh simulation. The instantaneous flow contours are also similar. The vortices at the trailing edge from the overset simulation are resolved better than those from the single mesh one. This is a consequence of the higher resolution of the overset meshes than the single mesh near the trailing edge (see Figure 5.19). Figure 5.22



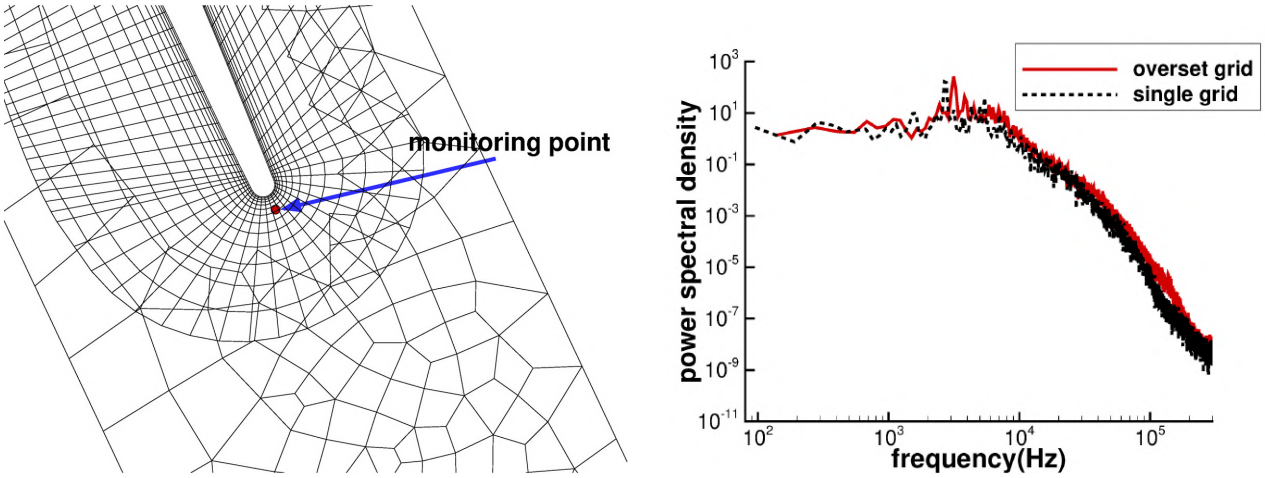

(a) pressure coefficient

(b) skin friction coefficient

Figure 5.22: The mean surface pressure coefficient and the mean skin friction coefficient of the T106A blade from overset and single mesh simulations. Experimental data is also compared with the numerical results.

shows the $c_p$ and $c_f$ profiles from overset and single mesh simulations. It is clearly seen that both $c_p$ and $c_f$ from overset meshes and the single mesh are almost on top of each other. The $c_p$ also

agrees with the experimental data (Stadtmüller, 2001) very well. To obtain the turbulence scales captured by these simulations, we also compare the power spectral density (PSD) of the pressure history at a very energetic point near the trailing edge (red dot in Figure 5.23(a)). The PSD is shown in Figure 5.23(b). Note that the PSD obtained from the overset mesh simulation agrees well with that from the single mesh simulation in the entire range of frequencies. This case serves as quantitative verification and validation of the high-order FR/CPR method for unsteady flow on stationary overset meshes.




(a) a monitoring point

(b) the power spectral density of pressure

Figure 5.23: A monitoring point (red dot) and the power spectral density of pressure.

## 5.5 Simulation of a rotor-stator flow using sliding meshes

Inspired by the benchmark T106A low-pressure turbine cascade case from the 4th International Workshop on High-Order Methods (https://how4.cenaero.be/), we designed a case with the turbine blade T106A under the wake of moving cylinders. The flow properties from this case are compared to the benchmark case, which has no cylinders. Figure 5.24 shows a 2D slice of the meshes for the simulations. The left side is a cylinder mesh and the right side is the T106A blade mesh. The diameter of each cylinder is 0.02 chord length of the blade. Five layers of elements with a length of 10% chord in total are extruded in the spanwise direction for each mesh. The cylinders

move upward in a constant speed corresponding to a Mach number 0.2935. A sliding interface exists between the two meshes with the mesh resolutions differing by a factor of 2 across the interface. The angle of the incoming flow with respect to the x-axis is 46.1°. The total pressure

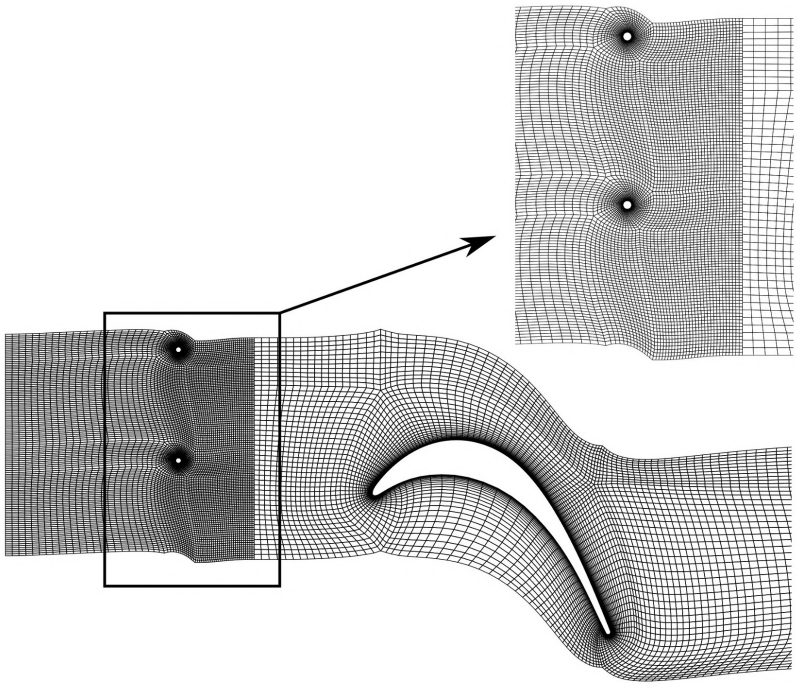
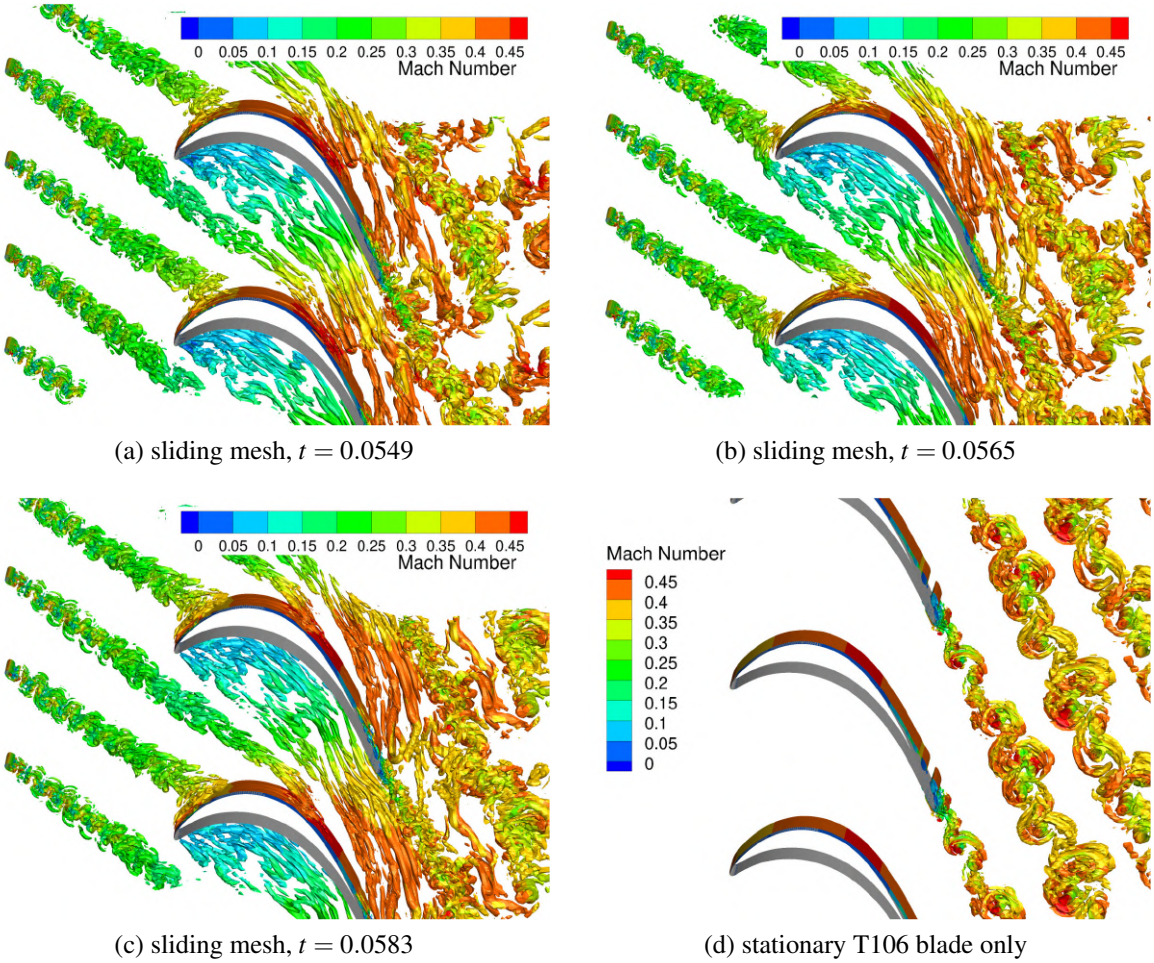


Figure 5.24: Meshes of two cylinders and a T106A turbine blade. A sliding interface exists between the two meshes.

and total temperature, as well as flow direction are imposed at the inlet, whereas static pressure is imposed at the outlet. At the exit the isentropic Mach number is 0.4 and the isentropic Reynolds number is 60,000 based on the chord length. Periodic boundary conditions are applied in both the spanwise and pitchwise directions. Data sampling begins at a time when several transients pass through the whole domain and the mean flows around the cylinders and the blade are sufficiently converged. The benchmark case only uses the single-zone blade mesh without cylinders. The boundary conditions and flow configurations of this case are exactly the same as the case with moving cylinders. To distinguish the two cases we denote the one with moving cylinders as sliding mesh case and the benchmark case as stationary case. Figure 5.25 shows the instantaneous iso-surfaces of the q-criterion colored by the Mach number for both case with $p = 2$. It is clear




(a) sliding mesh, $t = 0.0549$

(b) sliding mesh, $t = 0.0565$

(c) sliding mesh, $t = 0.0583$

(d) stationary T106 blade only

Figure 5.25: Iso-surfaces of Q-criterion for the simulation case of T106A blade behind moving cylinders. The iso-surfaces are colored by the Mach number. The benchmark case is also shown as comparison.

that a turbulent wake was generated by the moving cylinders, and these wakes seamlessly passed through the sliding interfaces without visible dissipation or distortion. Even though the mesh resolution across the sliding interface differs by a factor of two, no-spurious flow structures were generated by the interface, indicating that the interface treatment is satisfactory. At $t = 0.0549$ (Figure 5.25(a)), the wake vortices of the cylinder is about to touch the leading edge of the blade. Figure 5.25(b) shows the wake vortices of the cylinder is hitting the leading edge of the blade. The vortices are then split into two parts as shown in Figure 5.25(c). One part moves above the top surface of the blade (suction side), and the other part enters the region under the bottom surface of the blade (pressure side). Due to the different Mach number distribution, the vortices at the

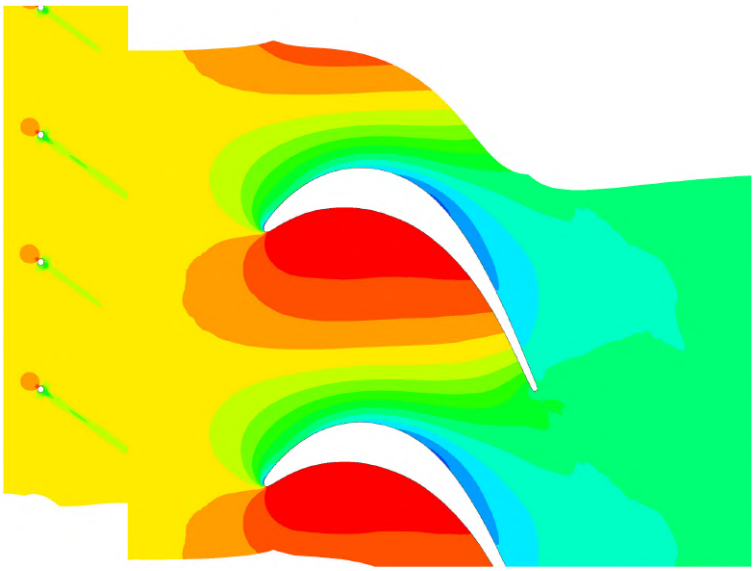
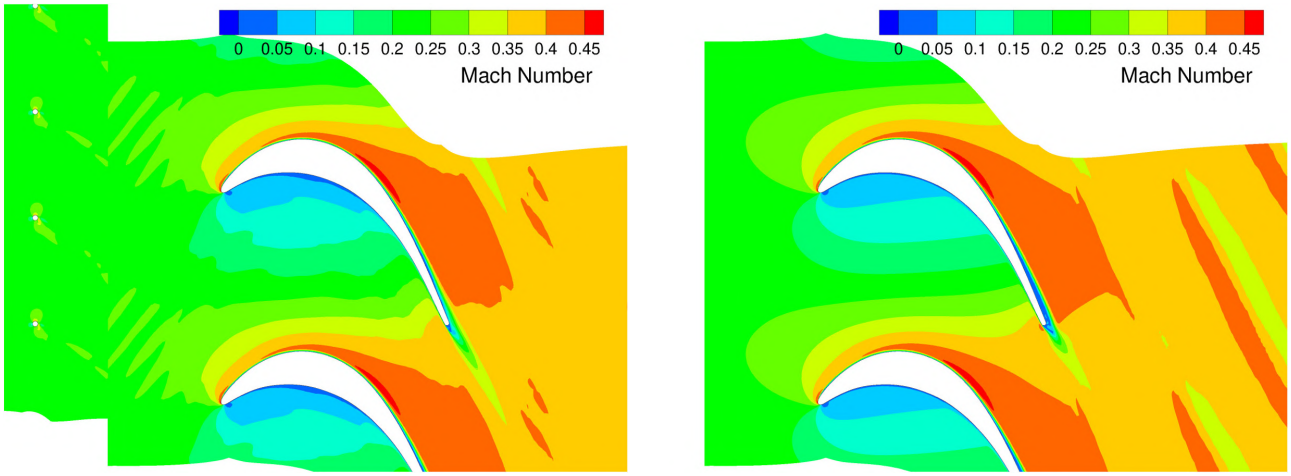


Figure 5.26: Mean pressure contours for the sliding mesh case.

pressure side moves much slower than the vortices at the suction side. The vortices are stretched by the sudden increase in flow speed. Once the wake hits the blade surface, visible turbulent spots appeared on the blade surface, indicating that the flow is significantly influenced by the wake. As seen in Figure 5.26, the pressure gradient above the blade is roughly aligned with the




(a) sliding mesh (b) stationary mesh

Figure 5.27: Mean Mach number contours around the T106A blade.

wall surface normal. This local pressure distribution "pushes" the wake vortices of the cylinders towards the wall surface. The pressure gradient under the blade is however opposite to the wall

surface normal. Vortices under the blade are therefore pushed away from the blade. After passing through the cascades, the vortices from both the suction side and the pressure side merge with the vortices generated at the trailing edge of the blade (Figure 5.25(a)-(c)). The cylinder wakes make the flow transition into a turbulent one earlier than without the wake. As a comparison in Figure 5.25(d) without the moving cylinders, the vortices after the trailing edge of the blades propagate downstream without much interactions. Figure 5.27 shows the contours of mean Mach number around the blades. The Mach number distribution of the sliding mesh case (Figure 5.27(a)) is very similar to that of the stationary mesh case (Figure 5.27(b)). However, differences appear at the pressure side of the blades. The contours of the sliding mesh case are less smooth than those of the stationary mesh case. This could be related to the nearly stationary vortices always exist in region at the pressure side (Figure 5.25(a)-(c)). To investigate the impact of the wake vortices

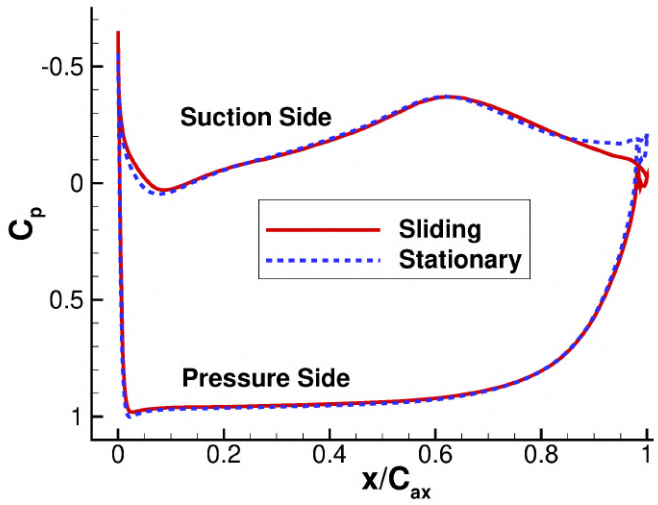
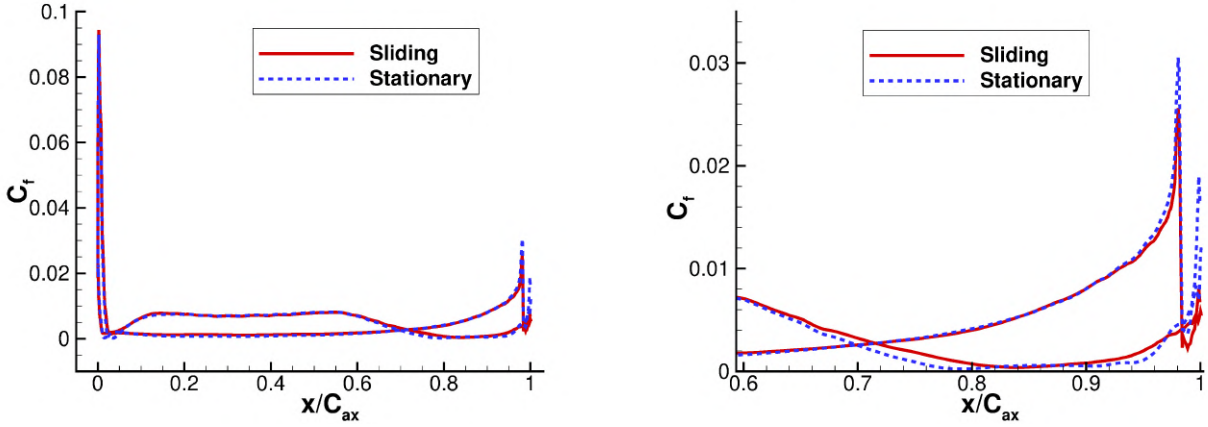



Figure 5.28: Mean surface pressure coefficient profiles. $C_{ax}$ is the axial chord length. The sliding mesh case is compared to the stationary mesh case.

of the moving cylinders on the pressure and friction on the wall surface of the blade, profiles of the mean pressure coefficient $c_p$ and the mean skin friction coefficient $c_f$ are shown in Figure 5.28 and 5.29. On pressure side the $c_p$ profile from the sliding mesh case is almost on top of that from the stationary mesh case. Significant difference appears on the suction side where vortices are generated from the blade. The $c_p$ in this region from the sliding mesh case is larger than




(a) overall view

(b) zoom in view near the trailing edge

Figure 5.29: Mean skin friction coefficient profiles. $C_{ax}$ is the axial chord length. The sliding mesh case is compared to the stationary mesh case.

the stationary mesh case. These observations indicates that, by vortex-structure interaction, the wake of the cylinders are not strong enough to change the mean pressure distribution on the blade. However, the mean pressure distribution is changed by the vortex-vortex interaction at the trailing edge. For $c_f$ in Figure 5.29(a), the profile from the sliding mesh case is on top of that from the stationary mesh on most part of the blade surface, with slight differs near the trailing edge. When zooming into the area close to the trailing edge (Figure 5.29(b)), it is seen that the $c_f$ from the sliding mesh case is significantly reduced at the trailing edge, and the separation point is closer to the trailing edge. This means the vortex-vortex interaction reduces the skin friction at the trailing edge and slightly shift the separation point to the downstream.

## 5.6 Viscous Flow over a Moving Airfoil

This case is also from the 4th International Workshop on High-Order CFD Methods (https://how4.cenaero.be). A NACA0012 airfoil undergoes various smooth flapping-type motions, starting from a position with a zero angle of attack. A sketch of the airfoil with the definition of geometrical parameters is shown in Figure 5.30. The center of rotation is at the $1/3$ chord from the leading edge. Three

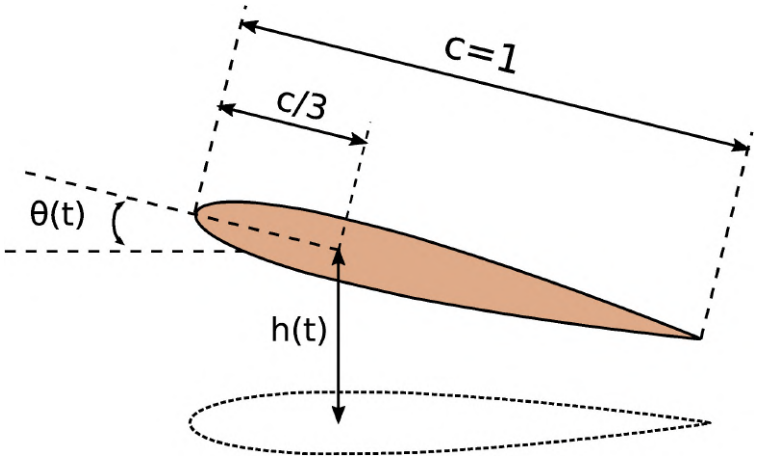


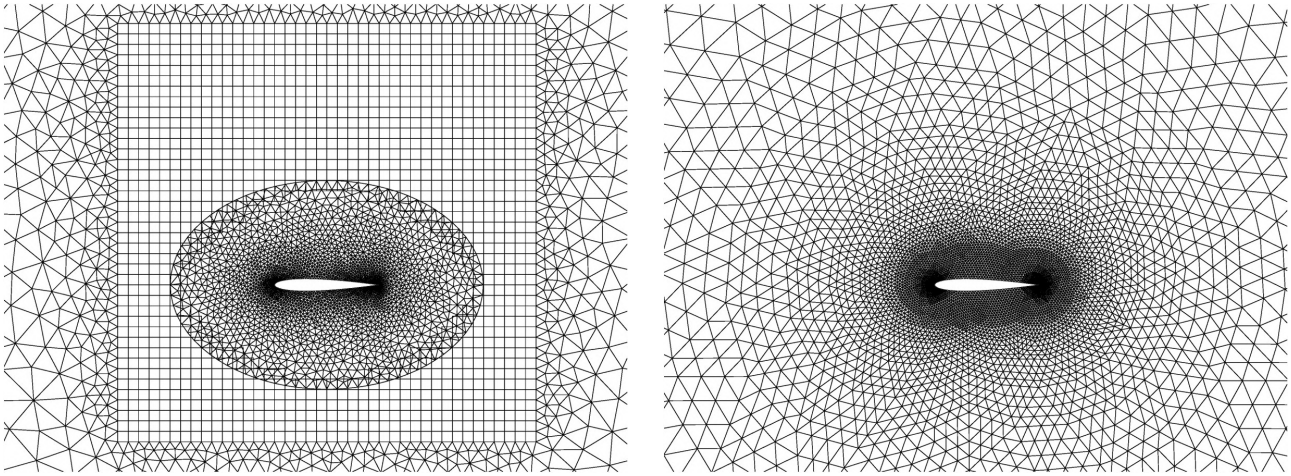

Figure 5.30: The definition of geometric parameters for the motions of a NACA0012 airfoil.

prescribed motions are considered:

$$\begin{aligned}
&\text{Pure heaving:} && h(t) = t^2(3-t)/4, \quad \theta(t) = 0 \\
&\text{Flow aligning:} && h(t) = t^2(3-t)/4, \quad \theta(t) = \pi t^2(t^2-4t+4)/3 \\
&\text{Energy extracting:} && h(t) = t^3(-8t^3+51t^2-111t+84)/16, \quad \theta(t) = 4\pi t^2(t^2-4t+4)/9
\end{aligned} \tag{5.4}$$

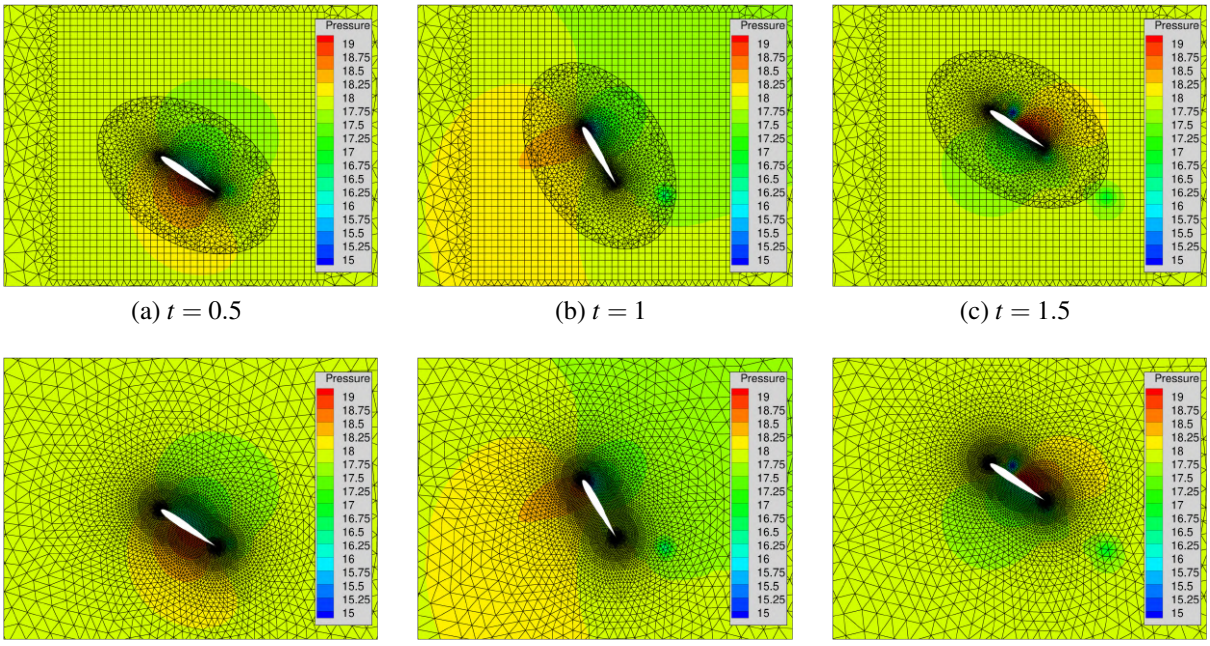
Each of the simulation starts from a steady solution at a zero angle of attack with an ending simulation time of $t = 2$. Both overset and non-overset (single zone) meshes are used for comparison purposes. The coarse mesh is refined twice to produce the medium and fine meshes for h-refinement studies. The medium meshes are shown in Figure 5.31. The single-zone meshes are downloaded from the High-Order CFD Workshop. At each level of refinement, the overset mesh has a similar resolution as the single-zone mesh. The far-field boundary is 100 chords away from the airfoil.

The flow conditions are set up as follows. The free-stream Mach number is $M_\infty = 0.2$, the initial angle of attack is zero, and the Reynolds number based on the chord is $Re = 1000$. To visualize the flow field, the pressure contours for $p = 2$ at $t$ =0.5, 1.0, and 1.5 are shown in Figure 5.32, 5.33, and 5.34 for the three types of motions on the medium overset and single zone meshes. Note that the overset and single-zone meshes produced qualitatively very similar flow solutions. To quantitatively verify the high-order flow solver for moving overset meshes, two physical quantities are computed to compare with other simulations. The first quantity is the work done by the fluid



(a) overset meshes (b) single zone mesh

Figure 5.31: The meshes with medium resolution for a NACA0012 airfoil.



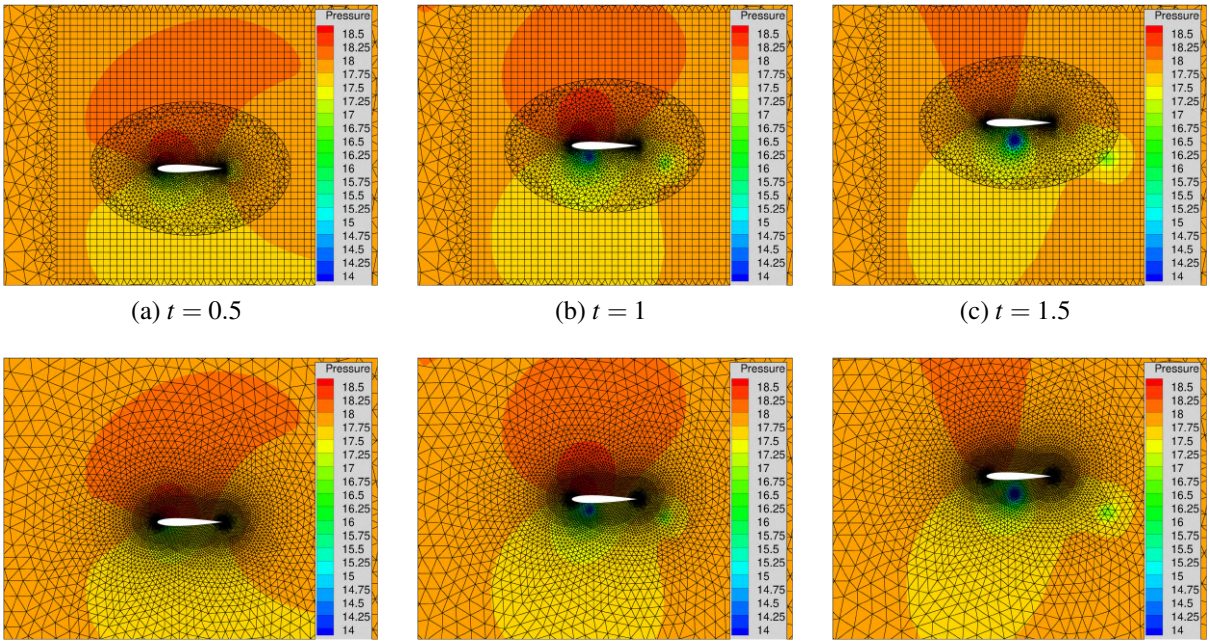

(a) $t = 0.5$ (b) $t = 1$ (c) $t = 1.5$

(d) $t = 0.5$ (e) $t = 1$ (f) $t = 1.5$

Figure 5.32: Pressure contours for the motion of flow aligning at $t$ =0.5, 1.0 and 1.5. The top row shows the overset while the bottom row shows the single-zone solutions. The p2 results on the medium meshes are shown.

force on the airfoil computed as

$$W = \int_0^T \oiint_{airfoil} \vec{v}_g(t) \cdot \vec{f}_{surf}(t) dsdt, \tag{5.5}$$




(a) $t = 0.5$ (b) $t = 1$ (c) $t = 1.5$

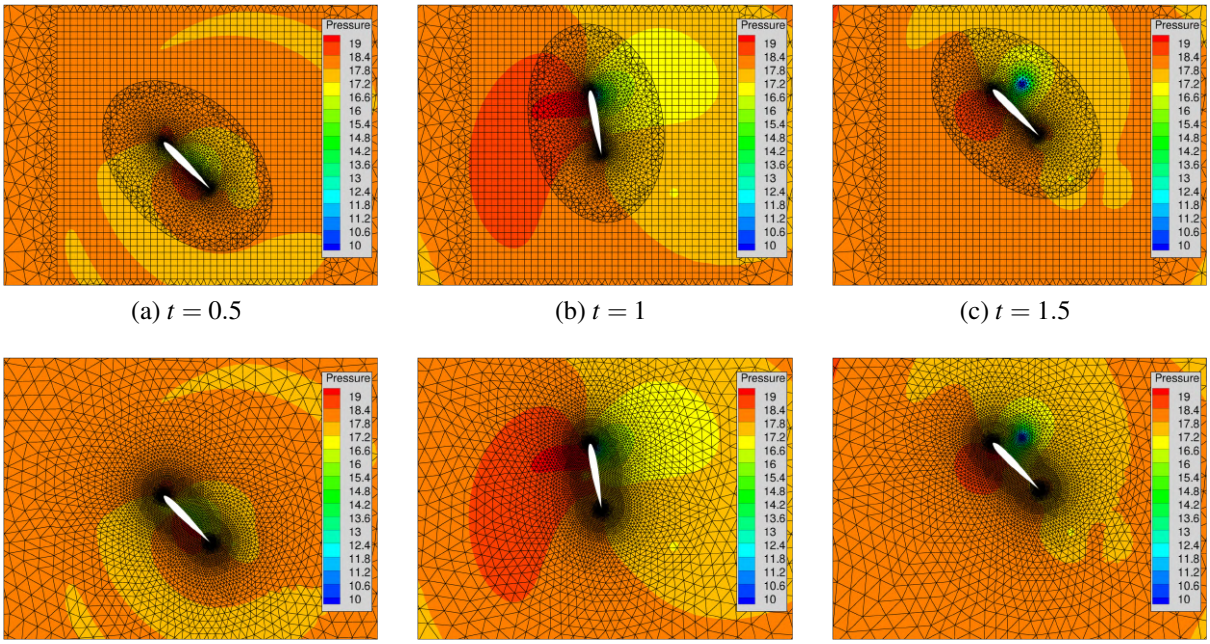
(d) $t = 0.5$ (e) $t = 1$ (f) $t = 1.5$

Figure 5.33: Contours of pressure for pure heaving at $t$ =0.5, 1.0 and 1.5. The top row shows the overset while the bottom row shows the single-zone solutions. The p2 results on the medium meshes are shown.

where $\vec{v}_g(t)$ is the grid velocity of the airfoil, $\vec{f}_{surf}(t)$ is the surface force vector, and T=2 is the ending simulation time. The other quantity is the vertical impulse (or y-impulse) from the fluid to the airfoil

$$I = \int_0^T F_y(t)dt. \tag{5.6}$$

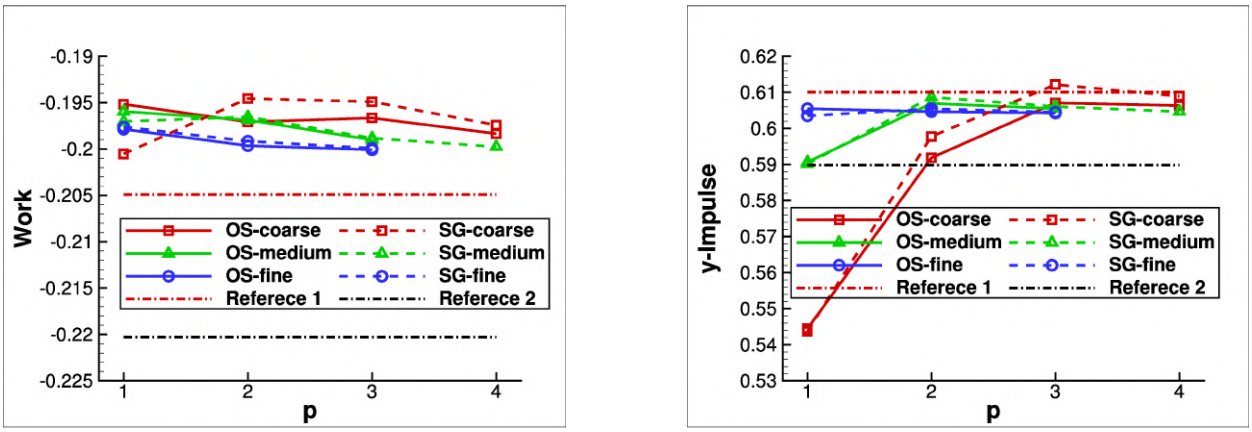
The same test cases were run on the single-zone meshes by two other groups from the University of California, Berkeley and the University of Michigan. Unfortunately the results from the two groups do not match each other. Therefore, we compare our computed work and y-impulse to the results from both groups. In order to demonstrate convergence in these quantities, we conducted an extensive hp-refinement study. Figure 5.35, 5.36 and 5.37 present the results on different meshes and with different p orders with comparison to the two groups. Note that for all three motions, the work changes significantly with $p$ on the coarse meshes (red squares). On the medium meshes, the difference of work becomes smaller with the change in $p$. On the fine meshes, this difference is even smaller. For example, the work for $p = 2$ and $p = 3$ is very close to each other (blue




(a) $t = 0.5$ (b) $t = 1$ (c) $t = 1.5$

(d) $t = 0.5$ (e) $t = 1$ (f) $t = 1.5$

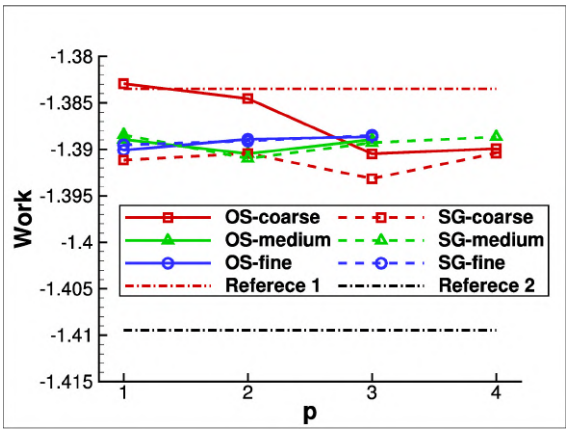
Figure 5.34: Contours of pressure for energy extracting at $t$ =0.5, 1.0 and 1.5. The top row shows the overset while the bottom row shows the single-zone solutions. The p2 results on the medium meshes are shown.



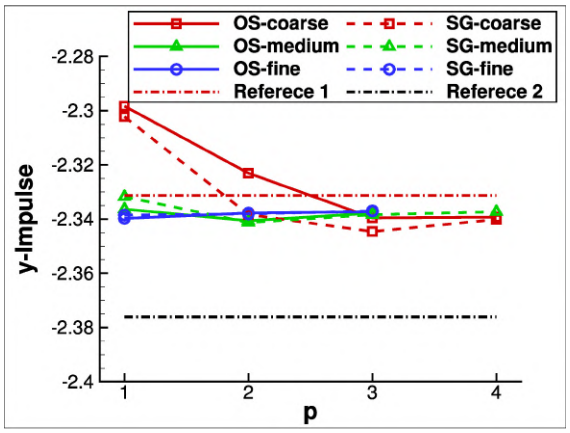

(a) work (b) impulse

Figure 5.35: Work and y-impulse for the flow aligning motion. "OS" denotes overset meshes and "SG" denotes single-zone meshes. Reference 1 is from a group in University of Michigan and reference 2 is from a group in University of California, Berkeley.

circles), demonstrating p-independence. For the fine meshes, the maximum percentage of the work difference between $p = 2$ and $p = 3$ is 0.38%, which is observed from flow aligning. This means the work is reaching p-convergence. For $p = 3$, the work from the medium meshes is almost

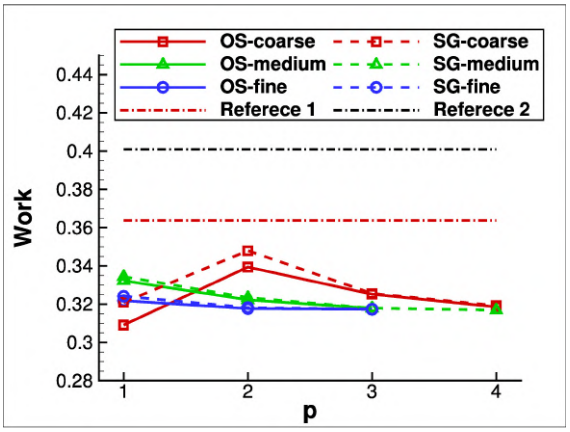



(a) work

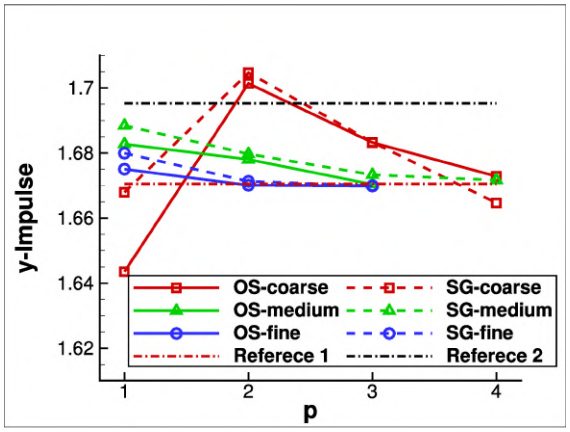



(b) impulse

Figure 5.36: Work and y-impulse for the pure heaving motion. "OS" denotes overset meshes and "SG" denotes single-zone meshes. Reference 1 is from a group in University of Michigan and reference 2 is from a group in University of California, Berkeley.




(a) work




(b) impulse

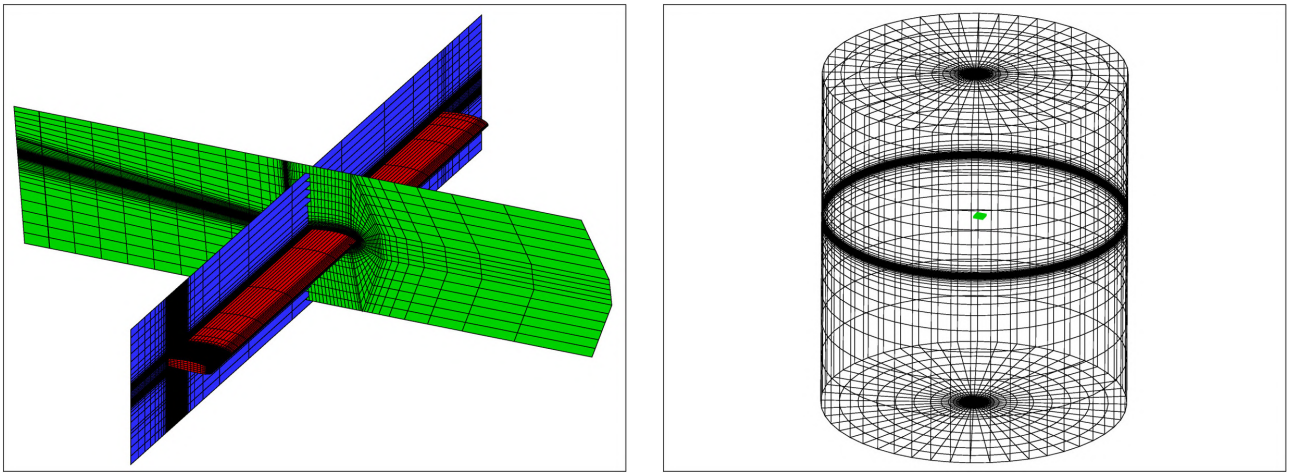
Figure 5.37: Work and y-impulse for the energy extracting motion. "OS" denotes overset meshes and "SG" denotes single-zone meshes. Reference 1 is from a group in University of Michigan and reference 2 is from a group in University of California, Berkeley.

on top of that from the fine meshes for all three motions. The maximum relative difference in work between the medium meshes and the fine meshes for $p = 3$ is 0.55%, which is again found with the flow aligning. This indicates that h-convergence is achieved. The same trend for the y-impulse can also be seen on the right of Figure 5.35-5.37. The p-convergence and h-convergence are both achieved on the fine meshes with a relative difference below 0.16% and 0.26% respectively. When comparing the difference of work and y-impulse between the overset meshes and single meshes

from the present high-order results ($p > 1$) , the percentage is generally less than 0.1% for all motions and all mesh resolutions except the coarse ones. When comparing to the results from the two groups, our converged values do not completely agree with either group. Our results are closer to those from the U. Michigan. However, the maximum difference is 12.7% for the work calculated from energy extracting (see Figure 5.37(a)). For more details on the simulation data from U. Michigan group and U.C. Berkeley group, please visit the resources: https://how4.cenaero.be, Presentation of case BL3.

## 5.7 Simulation of Single-bladed Hovering Rotor

Extensive experimental results for this case are documented in (Martin & Leishman, 2002). Therefore this case serves as a validation problem for the high-order overset solver. The airfoil section of the blade is the NACA2415 airfoil. The experiment used the baseline blade, which has no tapered or swept tip or twist. The aspect ratio is 9.12 with a 20% of root cutout. Overset background and blade meshes are employed in the present simulation. Figure 5.38(a) shows the wall surface mesh



(a) Blade mesh and cut planes in the near body mesh

(b) The background mesh and the near body mesh (green)

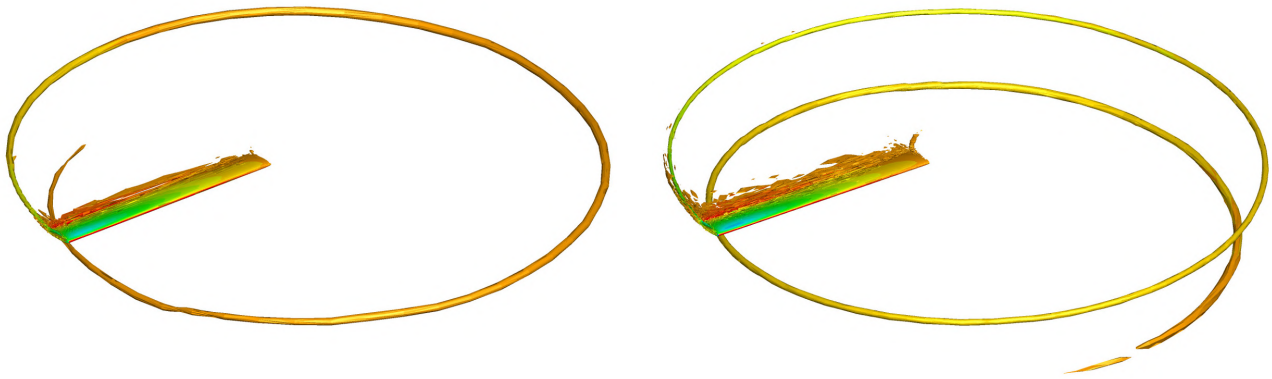
Figure 5.38: A near-body mesh for the hovering rotor and the background mesh.

of the blade (red) and two cut planes (blue and green) in the near body mesh. The mesh at the blade tip and the its down stream region is refined to resolve the tip vortex. The near body mesh is

9 chords in the streamwise direction and 2 chords in height. The size of the near body mesh allows the mesh resolution at the overset interface to match that of the background grid. A cylindrical background mesh is shown in Figure 5.38(b). The diameter of the background mesh is 30 times the rotor disk radius, and the height is 35 times of the rotor disk radius. To resolve the tip vortex, the mesh is refined in the region where it overlaps with the near body mesh. The refined section is about 3 chords in height. The rotor mesh has 267,785 elements, and the background mesh has 426,624 elements, resulting in a total of 5,555,272 DOFs/equation at p1 and 18,749,043 DOFs/equation at p2. A standard sea level condition with zero flow velocity is imposed on the boundary of the background mesh as the far field condition. The flow conditions are the same as those in the experiment (Martin & Leishman, 2002). The tip Mach number is to 0.26, the tip Reynolds number is $272,000$, and the blade angle of attack is $4.5°$. A summary of the rotor blade configuration and the flow conditions is given in Table 5.1. Staring from a quiescent air condition, implicit large sim-

Table 5.1: Parameters of the rotor blade.

| | | | | |
|---|---|---|---|---|
| Radius, $R$ | 406.0 mm | | Collective pitch, $\theta_0$ | $4.5°$ |
| Chord, $c$ | 44.5 mm | | Tip speed, $V_{tip}$ | $89.28m/s$ |
| Root cutout, $r_0$ | 20% | | Rotational frequency, $\Omega$ | 30 Hz |
| Airfoil section | NACA 2415 | | Tip Mach number, $M_{tip}$ | 0.26 |
| Twist, $\theta_{tw}$ | $0°$ | | Tip Reynolds number, $Re_{tip}$ | 272,000 |

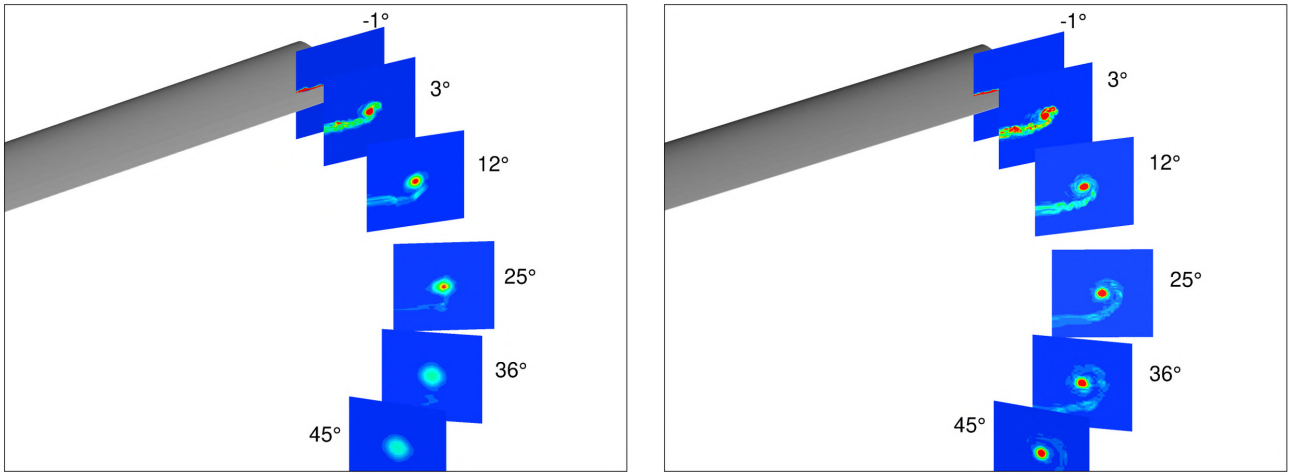


(a) $p = 1$ (b) $p = 2$

Figure 5.39: Iso-surfaces of the Q-criteria colored by pressure for the p1 and p2 simulations.

ulations were run for several revolutions until the tip vortex was fully developed and nearly steady with respect to the moving rotor. In total, 9 revolutions were run for the p1 simulation. Data was collected at the last 1.5 revolutions to compare with experimental results. Then the p2 simulation was restarted from the p1 simulation, and continued for 3 revolutions with the first 1.5 revolutions to develop the tip vortex, and the last 1.5 revolutions for data collection. Figure 5.39 shows the iso-surfaces of the q-criteria colored by pressure after the tip vortex was fully developed. Note that the $p = 1$ simulation resolves the tip vortex up to 400° after the trailing edge of the blade tip, while the $p = 2$ simulation preserves tip vortex for a longer distance, about 630° after the trailing edge. Because the mesh under the rotor is coarsened very quickly, the tip vortex is not preserved after two revolutions.

To provide a visual impression on the vortex core size, we plot the vorticity magnitude contours at various wake ages for both p1 and p2 simulations in Figure 5.40. The wake age is defined as




(a) $p = 1$ (b) $p = 2$

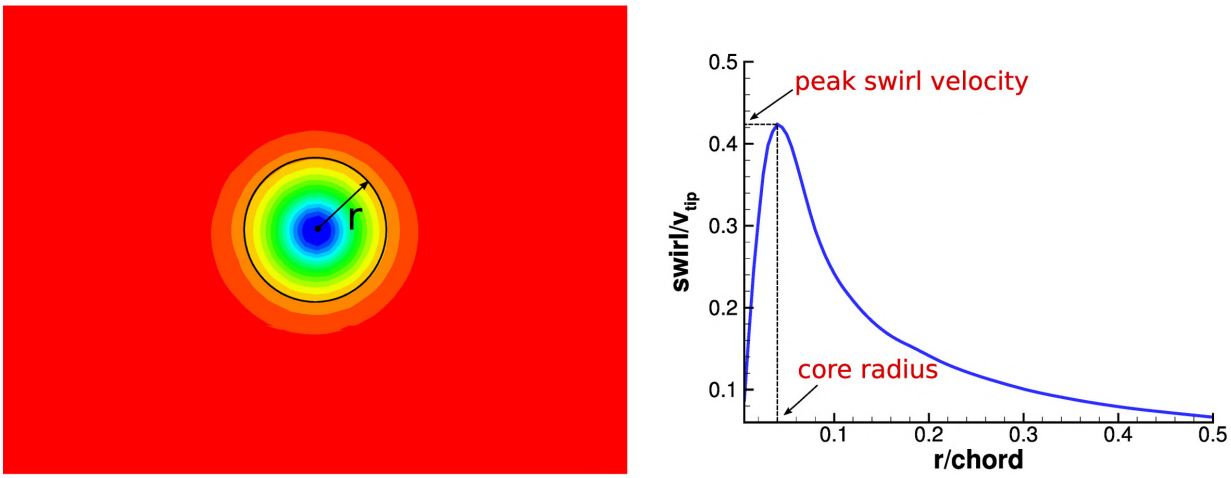
Figure 5.40: Contours of vorticity magnitude at different wake ages. The angle besides each slice is its wake age. The rotor blade is shown in grey.

the angle between the location of the vortex and the trailing edge of the blade, with respect to the rotational axis of the rotor. Note that the $p = 2$ simulation preserved the tip vortex much better than the p1 simulation, and also produced more flow structures around the tip vortex than the p1 simulation. Measurements in peak swirl velocity and core radius of the tip vortex at various wake

age were made in (Martin & Leishman, 2002). We also extracted the peak swirl velocity and core radius from the simulated flow field. At a given wake age, a slice of the flow field is plotted to identify the center of the vortex. A circle centered at the vortex core with radius $r$ is then used to compute the swirl velocity. Figure 5.41(a) illustrates a circle around the vortex core to compute the swirl velocity and the definition of the vortex core radius. We employ the density contours to identify the vortex core location. The swirl velocity $v_\theta$ at a given radius $r$ is computed as

$$v_\theta(r) = \frac{\oint_{C_r} \vec{v} \cdot d\vec{l}}{2\pi r}, \tag{5.7}$$

where $C_r$ is the circle, $\vec{v}$ is the flow velocity on the circle. The profile of the swirl velocity with respect to the radius is shown in Figure 5.41(b). The core radius of the vortex is found at the radius corresponding to the peak swirl velocity. Profiles of the swirl velocity at various wake ages are then used to compare with the measurements. Figure 5.42 shows both the computed and

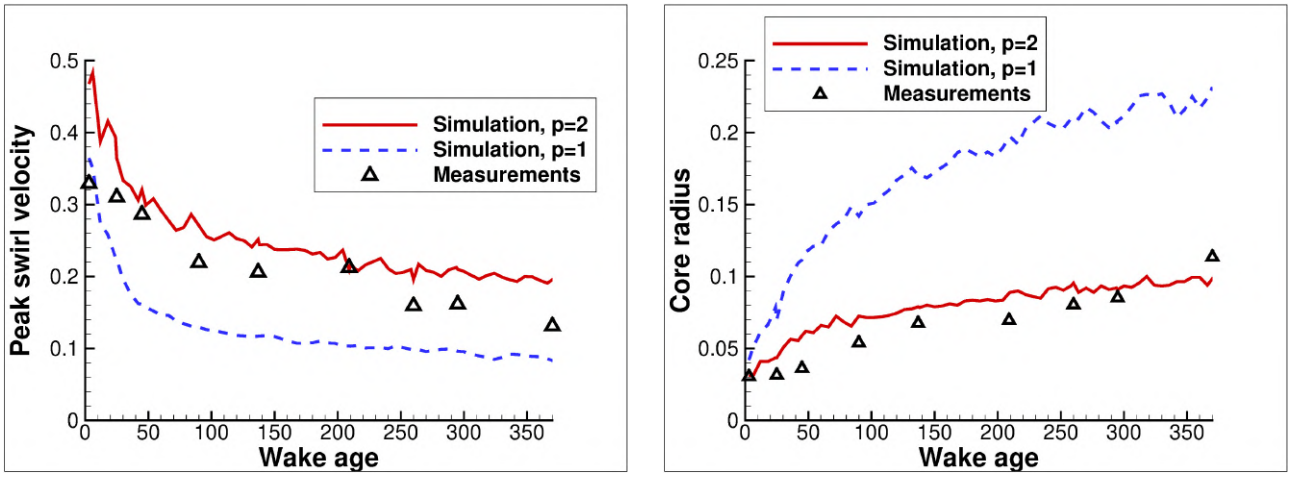



(a) a slice of tip vortex

(b) swirl velocity profile

Figure 5.41: Schematic for the computation of the vortex core swirl velocity. The radius is normalize by the chord length and the swirl velocity is normalize by the blade tip velocity.

measured peak swirl velocity and vortex core radius. Note that the $p = 2$ simulation produced much closer agreement with experimental data in the peak swirl velocity than the $p = 1$ simulation at each wake age except 3°. In Figure 5.42(b), it is seen that the core radius from the $p = 1$




(a) (b)

Figure 5.42: Peak swirl velocity and vortex core radius at different wake ages (in degrees). The peak swirl velocity is normalized by the blade tip velocity. The vortex core radius is normalize by the chord of the blade.

simulation grows so fast that it differs from the experimental data dramatically. In contrast, the core radius of the $p = 2$ simulation is close to the measurements at all the wake ages. These comparisons show that the $p2$ overset FR/CPR scheme is indeed much more accurate than the p1 scheme, and was capable of resolving the tip vortex in terms of the core radius and the peak swirl

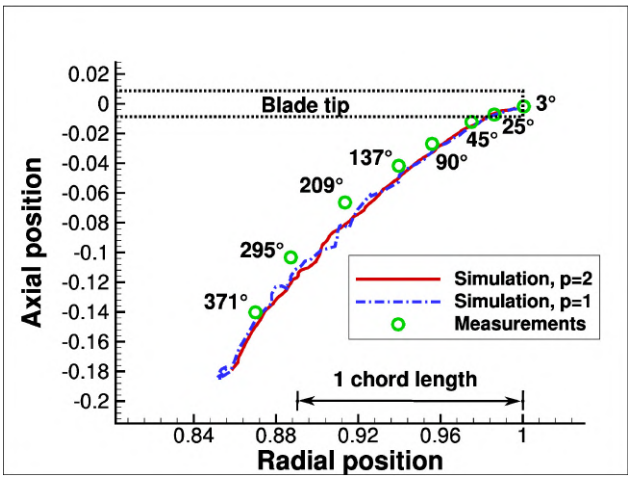



Figure 5.43: Tip vortex core location at different wake ages. The radial position is the distance between the vortex core and the rotational axis, and the axial position is the distance between the vortex core and the rotor disk. Both radial and axial positions are normalized by the rotor radius. The angles beside the experimental data are the wake ages.

velocity. Figure 5.43 compares the wake geometry between the computational and experimental data. It is known that for a hovering rotor, a wake boundary or slipstream exists in the flow below the rotor disk (Leishman, 2006). The flow outside the wake boundary is relatively quiescent while the flow inside this boundary moves downward due to the down wash of the rotor. Therefore a contraction is generated because of the difference in flow velocity. Since the tip vortex is convected along the wake boundary, the diameter of the wake decreases along with the wake age due to the contraction. In Figure 5.43, the radial and axial positions of the vortex core at various wake ages are plotted. Both the radial and axial positions are normalized by the rotor radius. The vortex position in the $p = 1$ simulation is almost the same as that in the $p = 2$ simulation, and agrees with the experimental measurement very well.

# Chapter 6

# Conclusions and Future Work

## 6.1 Summary and Conclusions

This dissertation presents the details of the implementation of a high-order FR/CPR method for overset/sliding meshes. The original FR/CPR method has been introduced in details. Parallel automatic hole-cutting and donor searching approaches for both static and moving grids have been implemented in a robust and efficient way to conduct real-world simulations. The donor searching approach is extended to handle the receptor-donor connectivity for sliding meshes. Both explicit and implicit schemes have been developed for time integration. Two types of data communication approaches, i.e., face-based and element-based interpolations, are compared. The face-based approach shows an advantage over the element-based approach in both stability and accuracy. Accuracy studies are performed with inviscid and viscous benchmark problems - the isentropic vortex and the Couette flow. The designed orders of accuracy for the cases with stationary/moving overset meshes and sliding meshes are obtained.

For stationary overset meshes, the overset FR/CPR method is demonstrated for both steady and unsteady turbulent flow problems. A flow over a sphere case with low-Reynolds number is simulated. Steady flow field is obtained. The computed streamlines behind the sphere are compared with experiment with good agreement. The profile of the skin friction coefficient is identical to that from another high-order method. The overset solver is then verified for unsteady flow by solving a transitional turbine cascade problem. The surface pressure coefficient agrees very well with experimental data. The skin friction coefficient and the power spectral density of pressure match numerical results from simulation with a single mesh.

For moving overset meshes, The high-order overset tool is used to simulate a benchmark moving grid problem from the 4th International Workshop on High-Order CFD Methods. Both overset and non-overset meshes are used in an extensive hp-refinement study. Three motions are imposed on a near body mesh of a NACA0012 airfoil. The work and vertical impulse are computed with different degrees of approximate polynomials and various mesh resolutions. Hp-convergence of these quantities are achieved. It is also verified that the overset and non-overset meshes produced nearly the same results, thus verifying the overset implementation. The work and vertical impulse are also compared with other groups from the workshop. As a special case of moving overset meshes, a rotor-stator flow problem is tackled using the sliding mesh approach. The rotors have a simple cylindrical shape, and generate turbulent wakes, which impinge the T106A turbine blades. The cylinders move in a constant Mach number of 0.2935. The interactions between the cylinder vortices and the blade were briefly analyzed. The mean Mach number distribution near the pressure side of the turbine blade is disturbed by the cylinder wakes. Comparing to a benchmark case without the moving cylinders, both the pressure coefficient and the skin friction coefficient at the trailing edge of the blade change significantly due to the interaction between the cylinder wake and the blade vortices. Finally the solver is validated for a hovering rotor problem. The tip vortex is resolved well by the high-order simulation at p2. The peak swirl velocity and vortex core radius are compared with experimental data. Results from $p = 2$ simulation shows good agreement with experiment data. The wake geometry resolved from both the $p = 2$ and $p = 1$ simulations is very close to experimental data.

## 6.2 Future Work

While this dissertation provides high-order results which agree well with the results from literatures and experiments, some potential future work directions are outline below:

1. **Treatment for Strong Discontinuities in Flow Field**

   None of the test cases in this dissertation involve shock waves in the flow. However, strong

discontinuities exist in a lot of flow phenomena, such as shock wakes, expansion waves and contact discontinuities. The range of applications of the high-order overset FR/CPR method would be greatly expanded if those discontinuities can be handled.

2. **Non-conformal Periodic Boundary Interfaces**

   In the sliding meshes, the boundary surfaces at the sliding interface are non-conformal. Note that the use of non-conformal interfaces can also simplify the mesh generation for simulations with periodic boundary conditions, which are quite common in turbo-machinery problems. It is nontrivial to generate three-dimensional structured meshes with conformal periodic boundary interfaces for complex geometries. Even for unstructured 3D meshes, special cares need to be taken for periodic boundary interfaces. When high-order meshes are needed, which is often the case for high-order methods, generating a mesh with conformal periodic boundary interfaces becomes even more challenge. The using of non-conformal periodic interfaces can relieve this difficulty. Similar to the treatment for sliding interfaces, receptor-donor connectivity between the periodic boundaries can be built after shifting the mesh by one period in space.

3. **Adaptation**

   In a simulation of a rotor, while benefiting from the overset method with moving bodies, the capability to resolve the tip vortices is still limited by the local mesh resolution where the vortices pass through. It is not practical to refine the mesh globally. Using hp-adaptation can provide more resolution at the location where the tip vortices propagate through. Therefore the adaptive capability of the high-order overset FR/CPR method would greatly boost its utility in the simulations of rotors or moving bodies with turbulent wake.

# References


Ahmad, J. & Duque, E. P. N. (1996). Helicopter rotor blade computation in unsteady flows using moving overset grids. *Journal of Aircraft*, 33(1), 54–60.

Arfken, G. B. (1985). Gibbs phenomenon. In *Mathematical Methods for Physicists, 3rd Edition* chapter 14, (pp. 783–787). Orlando, FL: Academic Press.

Argyris, J. H. & Kelsey, S. (1960). *Energy theorems and structural analysis*, volume 60. Springer.

Arnold, D. N., Brezzi, F., Cockburn, B., & Marini, L. D. (2002). Unified analysis of discontinuous galerkin methods for elliptic problems. *SIAM journal on numerical analysis*, 39(5), 1749–1779.

Bakker, A., LaRoche, R. D., Wang, M.-H., & Calabrese, R. V. (1997). Sliding mesh simulation of laminar flow in stirred reactors. *Chemical Engineering Research and Design*, 75(1), 42–44.

Bassi, F. & Rebay, S. (1997). A high-order accurate discontinuous finite element method for the numerical solution of the compressible navier–stokes equations. *Journal of computational physics*, 131(2), 267–279.

Bassi, F. & Rebay, S. (2000). Gmres discontinuous galerkin solution of the compressible navier-stokes equations. In *Discontinuous Galerkin Methods* (pp. 197–208). Springer.

Benek, J., Buning, P., & Steger, J. (1985). A 3-d chimera grid embedding technique. In *7th Computational Physics Conference* (pp. 1523).

Bonet, J. & Peraire, J. (1991). An alternating digital tree (adt) algorithm for 3d geometric searching and intersection problems. *International Journal for Numerical Methods in Engineering*, 31(1), 1–17.

Brazell, M. J., Sitaraman, J., & Mavriplis, D. J. (2016). An overset mesh approach for 3d mixed element high-order discretizations. *Journal of Computational Physics*, 322, 33–51.

Brown, D., Henshaw, W., & Quinlan, D. (1999). Overture-object-oriented tools for overset grid applications. In *17th Applied Aerodynamics Conference* (pp. 3130).

Chen, R. & Wang, Z. (2000). Fast, block lower-upper symmetric gauss-seidel scheme for arbitrary grids. *AIAA journal*, 38(12), 2238–2245.

Clough, R. W. (1960). The finite element method in plane stress analysis. In *Proceedings of 2nd ASCE Conference on Electronic Computation, Pittsburgh Pa., Sept. 8 and 9, 1960* (pp. 345–378).

Cockburn, B., Karniadakis, G. E., & Shu, C.-W. (2000). The development of discontinuous galerkin methods. In *Discontinuous Galerkin Methods* (pp. 3–50). Springer.

Cockburn, B. & Shu, C.-W. (1998). The runge–kutta discontinuous galerkin method for conservation laws v: multidimensional systems. *Journal of Computational Physics*, 141(2), 199–224.

Crabill, J., Witherden, F. D., & Jameson, A. (2018). A parallel direct cut algorithm for high-order overset methods with application to a spinning golf ball. *Journal of Computational Physics*, 374, 692–723.

Crabill, J. A., Sitaraman, J., & Jameson, A. (2016a). A high-order overset method on moving and deforming grids. In *AIAA Modeling and Simulation Technologies Conference* (pp. 3225).

Crabill, J. A., Sitaraman, J., & Jameson, A. (2016b). A high-order overset method on moving and deforming grids. In *AIAA Modeling and Simulation Technologies Conference* (pp. 3225).

Delanaye, M. & Liu, Y. (1999). Quadratic reconstruction finite volume schemes on 3d arbitrary unstructured polyhedral grids. In *14th Computational Fluid Dynamics Conference* (pp. 3259).

Dolejší, V. (2004). On the discontinuous galerkin method for the numerical solution of the navier–stokes equations. *International Journal for Numerical Methods in Fluids*, 45(10), 1083–1106.

Duan, Z. & Wang, Z. J. (2019). A high order overset fr/cpr method for dynamic moving grids. In *AIAA Scitech 2019 Forum* (pp. 1399).

Ferrer, E. & Willden, R. H. (2012). A high order discontinuous galerkin–fourier incompressible 3d navier–stokes solver with rotating sliding meshes. *Journal of Computational Physics*, 231(21), 7037–7056.

Fillola, G., Le Pape, M.-C., & Montagnac, M. (2004). Numerical simulations around wing control surfaces. In *24th International Congress of the Aeronautical Sciences ICAS*.

Francois, B., Costes, M., Dufour, G., Cerfacs, C., & France, T. (2011). Comparison of chimera and sliding mesh techniques for unsteady simulations of counter rotating open-rotors. In *20th ISABE Conference*.

Fujii, K. (1995). Unified zonal method based on the fortified solution algorithm. *Journal of Computational Physics*, 118(1), 92–108.

Galbraith, M. C. (2013). *A Discontinuous Galerkin Chimera Overset Solver*. PhD thesis, University of Cincinnati.

Galbraith, M. C., Benek, J. A., Orkwis, P. D., & Turner, M. G. (2015). A discontinuous galerkin scheme for chimera overset viscous meshes on curved geometries. *Computers & Fluids*, 119, 176–196.

Geuzaine, P., Grandmont, C., & Farhat, C. (2003). Design and analysis of ale schemes with provable second-order time-accuracy for inviscid and viscous flow simulations. *Journal of Computational Physics*, 191(1), 206–227.

Godunov, S. (1959). A finite difference method for the computation of discontinuous solutions of the equations of fluid dynamics. *Sbornik: Mathematics*, 47(8-9), 357–393.

Gottschalk, S., Lin, M., Manocha, D., & Tree, O. (1996). A hierarchical structure for rapid interference detection. In *Proceedings of ACM Siggraph*.

Haga, T., Gao, H., & Wang, Z. J. (2011). A high-order unifying discontinuous formulation for the navier-stokes equations on 3d mixed grids. *Mathematical Modelling of Natural Phenomena*, 6(3), 28–56.

Harten, A., Engquist, B., Osher, S., & Chakravarthy, S. R. (1987). Uniformly high order accurate essentially non-oscillatory schemes, iii. In *Upwind and high-resolution schemes* (pp. 218–290). Springer.

Hartmann, R. & Houston, P. (2006). Symmetric interior penalty dg methods for the compressible navier-stokes equations i: Method formulation. *International Journal of Numerical Analysis and Modeling*, 3(1), 1–20.

Henshaw, W. D. (1994). A fourth-order accurate method for the incompressible navier-stokes equations on overlapping grids. *Journal of computational physics*, 113(1), 13–25.

Hildebrand, F. B. (1987). *Introduction to numerical analysis*. Courier Corporation.

Holst, T. & Pulliam, T. (2010). Optimization of overset solution adaptive grids for hovering rotorcraft flows. In *AHS Aeromechanics Specialist's Conference*.

Hughes, T. J. (2012). *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation.

Huynh, H. T. (2007). A flux reconstruction approach to high-order schemes including discontinuous galerkin methods. In *18th AIAA Computational Fluid Dynamics Conference* (pp. 4079).

Huynh, H. T. (2009). A reconstruction approach to high-order schemnes including discontinuous galerkin for diffusion. In *47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition* (pp. 403).

Ims, J., Duan, Z., & Wang, Z. J. (2015). meshcurve: an automated low-order to high-order mesh generator. In *22nd AIAA computational fluid dynamics conference* (pp. 2293).

Jameson, A. & Yoon, S. (1987). Lower-upper implicit schemes with multiple grids for the euler equations. *AIAA journal*, 25(7), 929–935.

Jia, F., Wang, Z., Bhaskaran, R., Paliath, U., & Laskowski, G. M. (2019). Accuracy, efficiency and scalability of explicit and implicit fr/cpr schemes in large eddy simulation. *Computers & Fluids*, 195, 104316.

Katz, A., Wissink, A. M., Sankaran, V., Meakin, R. L., & Chan, W. M. (2010). Application of strand meshes to complex aerodynamic flowfields. In *28th AIAA Applied Aerodynamics Conference* (pp. 4934).

Klaij, C. M., van der Vegt, J. J., & van der Ven, H. (2006). Space–time discontinuous galerkin method for the compressible navier–stokes equations. *Journal of Computational Physics*, 217(2), 589–611.

Lakshminarayan, V. K., Sitaraman, J., & Wissink, A. M. (2017). Sensitivity of rotorcraft hover predictions to mesh resolution in strand grid framework. In *55th AIAA Aerospace Sciences Meeting* (pp. 1672).

Lee, Y. & Baeder, J. (2003). Implicit hole cutting-a new approach to overset grid connectivity. In *16th AIAA Computational Fluid Dynamics Conference* (pp. 4128).

Legras, G., Gourdain, N., Sicot, F., & Roumeas, M. (2009). Time spectral calculation of a casing treatment configuration for a high pressure compressor. In *8th European Conference on Turbomachinery Fluid Dynamics and Thermodynamics*.

Leishman, G. J. (2006). *Principles of helicopter aerodynamics with CD extra*. Cambridge university press.

Lele, S. K. (1992). Compact finite difference schemes with spectral-like resolution. *Journal of computational physics*, 103(1), 16–42.

Lim, J. W. & Strawn, R. C. (2008). Computational modeling of hart ii blade-vortex interaction loading and wake system. *Journal of Aircraft*, 45(3), 923–933.

Liu, X.-D., Osher, S., & Chan, T. (1994). Weighted essentially non-oscillatory schemes. *Journal of computational physics*, 115(1), 200–212.

Liu, Y., Vinokur, M., & Wang, Z. J. (2006). Spectral difference method for unstructured grids i: basic formulation. *Journal of Computational Physics*, 216(2), 780–801.

Loehner, R., Sharov, D., Luo, H., & Ramamurti, R. (2001). Overlapping unstructured grids. In *39th Aerospace Sciences Meeting and Exhibit* (pp. 439).

Martin, P. B. & Leishman, J. G. (2002). Trailing vortex measurements in the wake of a hovering rotor blade with various tip shapes. In *58th annual forum of the American Helicopter Society*.

Massing, A., Larson, M. G., & Logg, A. (2013). Efficient implementation of finite element methods on nonmatching and overlapping meshes in three dimensions. *SIAM Journal on Scientific Computing*, 35(1), C23–C47.

Mavriplis, D. J. (2002). An assessment of linear versus nonlinear multigrid methods for unstructured mesh solvers. *Journal of Computational Physics*, 175(1), 302–325.

McNaughton, J., Afgan, I., Apsley, D., Rolfo, S., Stallard, T., & Stansby, P. (2014). A simple sliding-mesh interface procedure and its application to the cfd simulation of a tidal-stream turbine. *International journal for numerical methods in fluids*, 74(4), 250–269.

MEAKIN, R. (1993). Moving body overset grid methods for complete aircraft tiltrotor simulations. In *11th Computational Fluid Dynamics Conference* (pp. 3350).

Miller, S. T., Campbell, R., Elsworth, C., Pitt, J., & Boger, D. (2014). An overset grid method for fluid-structure interaction. *World Journal of Mechanics*, 4(07), 217.

Nakahashi, K., Togashi, F., & Sharov, D. (2000). Intergrid-boundary definition method for overset unstructured grid approach. *AIAA journal*, 38(11), 2077–2084.

Nastase, C., Mavriplis, D., & Sitaraman, J. (2011). An overset unstructured mesh discontinuous galerkin approach for aerodynamic problems. In *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition* (pp. 195).

Noack, R. (2005). Suggar: a general capability for moving body overset grid assembly. In *17th AIAA computational fluid dynamics conference* (pp. 5117).

Noack, R. (2007). A direct cut approach for overset hole cutting. In *18th AIAA Computational Fluid Dynamics Conference* (pp. 3835).

Pärt-Enander, E. & Sjögreen, B. (1994). Conservative and non-conservative interpolation between overlapping grids for finite volume solutions of hyperbolic problems. *Computers & fluids*, 23(3), 551–574.

Peraire, J. & Persson, P.-O. (2008). The compact discontinuous galerkin (cdg) method for elliptic problems. *SIAM Journal on Scientific Computing*, 30(4), 1806–1824.

Rai, M. M. (1986). A conservative treatment of zonal boundaries for euler equation calculations. *Journal of Computational Physics*, 62(2), 472–503.

Rockwell, D. (1998). Vortex-body interactions. *Annual review of fluid mechanics*, 30(1), 199–229.

Roe, P. L. (1981). Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of computational physics*, 43(2), 357–372.

Rogers, S. E., Suhs, N. E., & Dietz, W. E. (2003). Pegasus 5: an automated preprocessor for overset-grid computational fluid dynamics. *AIAA journal*, 41(6), 1037–1045.

Roget, B. & Sitaraman, J. (2014). Robust and efficient overset grid assembly for partitioned unstructured meshes. *Journal of Computational Physics*, 260(1), 1–24.

Rusanov, V. V. (1962). *Calculation of interaction of non-steady shock waves with obstacles*. NRC, Division of Mechanical Engineering.

Saad, Y. (2003). *Iterative methods for sparse linear systems*, volume 82. siam.

Sankaran, V., Wissink, A., Datta, A., Sitaraman, J., Potsdam, M., Jayaraman, B., Katz, A., Kamkar, S., Roget, B., Mavriplis, D., et al. (2011). Overview of the helios version 2.0 computational platform for rotorcraft simulations. In *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition* (pp. 1105).

Shu, C.-W. (2014). Discontinuous galerkin method for time-dependent problems: survey and recent developments. In *Recent developments in discontinuous Galerkin finite element methods for partial differential equations* (pp. 25–62). Springer.

Sitaraman, J., Floros, M., Wissink, A., & Potsdam, M. (2010). Parallel domain connectivity algorithm for unsteady flow computations using overlapping and adaptive grids. *Journal of Computational Physics*, 229(12), 4703–4723.

Sitaraman, J., Lakshminarayan, V. K., Roget, B., & Wissink, A. M. (2017). Progress in strand mesh generation and domain connectivity for dual-mesh cfd simulations. In *55th AIAA Aerospace Sciences Meeting* (pp. 0288).

Spiegel, S. C., Huynh, H., & DeBonis, J. R. (2015). A survey of the isentropic euler vortex problem using high-order methods. In *22nd AIAA Computational Fluid Dynamics Conference* (pp. 2444).

Stadtmüller, P. (2001). Investigation of wake-induced transition on the lp turbine cascade t106 a-eiz. *DFG-Verbundprojekt Fo*, 136(11).

Stangl, R. & Wagner, S. (1996). Euler simulation of a helicopter configuration in forward flight using a chimera technique. In *52nd Annual American Helicopter Society Forum*.

Steger, J. L., Dougherty, F. C., & Benek, J. A. (1983). A chimera grid scheme.[multiple overset body-conforming mesh system for finite difference adaptation to complex aircraft configurations]. In *American Society of Mechanical Engineers*, volume 5 (pp. 59–69).

Steijl, R. & Barakos, G. (2008). Sliding mesh algorithm for cfd analysis of helicopter rotor–fuselage aerodynamics. *International journal for numerical methods in fluids*, 58(5), 527–549.

Strawn, R. C., Caradonna, F. X., & Duque, E. P. (2006). 30 years of rotorcraft computational fluid dynamics research and development. *Journal of the American Helicopter Society*, 51(1), 5–21.

Strawn, R. C., Kenwright, D. N., & Ahmad, J. (1999). Computer visualization of vortex wake systems. *AIAA journal*, 37(4), 511–512.

Sun, Y., Wang, Z. J., & Liu, Y. (2007). High-order multidomain spectral difference method for the navier-stokes equations on unstructured hexahedral grids. *Communications in Computational Physics*, 2(2), 310–333.

Taneda, S. (1956). Experimental investigation of the wake behind a sphere at low reynolds numbers. *Journal of the Physical Society of Japan*, 11(10), 1104–1108.

Thomas, J. W. (2013). *Numerical partial differential equations: finite difference methods*, volume 22. Springer Science & Business Media.

Toro, E. F. (2013). *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*. Springer Science & Business Media.

Turner, M. (1956). Stiffness and deflection analysis of complex structures. *journal of the Aeronautical Sciences*, 23(9), 805–823.

Van den Abeele, K., Lacor, C., & Wang, Z. J. (2008). On the stability and accuracy of the spectral difference method. *Journal of Scientific Computing*, 37(2), 162–188.

Van Leer, B. (1979). Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov's method. *Journal of computational Physics*, 32(1), 101–136.

Van Leer, B. (1985). Upwind-difference methods for aerodynamic problems governed by the euler equations. *Lectures in applied mathematics*, 22(Part 2), 327–336.

Vincent, P. E., Castonguay, P., & Jameson, A. (2011). A new class of high-order energy stable flux reconstruction schemes. *Journal of Scientific Computing*, 47(1), 50–72.

Visbal, M. R. & Gaitonde, D. V. (2002). On the use of higher-order finite-difference schemes on curvilinear and deforming meshes. *Journal of Computational Physics*, 181(1), 155–185.

Wang, Z. (1998). A conservative interface algorithm for moving chimera (overlapped) grids. *International Journal of Computational Fluid Dynamics*, 10(3), 255–265.

Wang, Z. J., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H. T., et al. (2013). High-order cfd methods: current status and perspective. *International Journal for Numerical Methods in Fluids*, 72(8), 811–845.

Wang, Z. J. & Gao, H. (2009). A unifying lifting collocation penalty formulation including the discontinuous galerkin, spectral volume/difference methods for conservation laws on mixed grids. *Journal of Computational Physics*, 228(21), 8161–8186.

Wang, Z. J. & Liu, Y. (2002). Spectral (finite) volume method for conservation laws on unstructured grids: Ii. extension to two-dimensional scalar equation. *Journal of Computational Physics*, 179(2), 665–697.

Wang, Z. J., Liu, Y., May, G., & Jameson, A. (2007). Spectral difference method for unstructured grids ii: extension to the euler equations. *Journal of Scientific Computing*, 32(1), 45–71.

Yee, H. C., Sandham, N. D., & Djomehri, M. J. (1999). Low-dissipative high-order shock-capturing methods using characteristic-based filters. *Journal of computational physics*, 150(1), 199–238.

Zhang, B. & Liang, C. (2015a). A simple, efficient, and high-order accurate curved sliding-mesh interface approach to spectral difference method on coupled rotating and stationary domains. *Journal of Computational Physics*, 295, 147–160.

Zhang, B. & Liang, C. (2015b). A simple, efficient, high-order accurate sliding-mesh interface approach to fr/cpr method on coupled rotating and stationary domains. In *53rd AIAA Aerospace Sciences Meeting* (pp. 1742).

Zhang, B., Liang, C., Yang, J., & Rong, Y. (2016). A 2d parallel high-order sliding and deforming spectral difference method. *Computers & Fluids*, 139, 184–196.

Zhang, M. & Shu, C.-W. (2005). An analysis of and a comparison between the discontinuous galerkin and the spectral finite volume methods. *Computers & fluids*, 34(4-5), 581–592.

Zienkiewicz, O. C., Taylor, R. L., Nithiarasu, P., & Zhu, J. (1977). *The finite element method*, volume 3. McGraw-hill London.